基于图论对于快速撤离问题的处理

方泽雄　胡诗敏　易钰洁

摘要

快速撤离问题是指，对于一栋复杂的大型建筑，当发生突发问题时，如何安排其中的人员快速撤离建筑物。这就涉及到人员撤离路径的选择。关于此问题，本文采用用图论模型，将一栋建筑物转化为图，包含节点和连接节点的弧。其中节点为建筑物内道路的交汇点以及受灾点（需要疏散人员的地点）和出口，而连接这些节点的弧在实际中则是建筑物内的路。每个人撤离则是要选择一条由受灾点到出口的路径。在这个问题中，每个人在其逃生路径上不一定能够一直通畅地走到终点，因为每段弧每秒种的人流量有一个最大值，很多人在同一时刻到达一段弧时，可能会有停滞现象。所以要整体地安排每个人的疏散路径，确保每段弧每个时刻的人流量都不超过它的最大人流量，在这个条件下，找到安排方案，使得所有人撤离的时间最短。

对于这个模型，可以用两种方法求解。一个是利用动态网络流的思想。由于每个人出发时间相同，逃生路径最短如果沿着最短路径会首先到达终点，故可以先饱和疏散最短路径上受灾点的人。基于此思想，将整个疏散过程看作是一轮一轮的疏散过程。第一轮先计划饱和疏散最短路径（指所有受灾点到所有出口的路径中的最短路径）上受灾点的人，计划好一条路后就更新路上涉及弧段的容量，重复此步骤，直到图中没有连接受灾点和出口的路，然后开始第一轮疏散，将一个受灾点被疏散完毕记为第一轮结束。此时将有关弧段的容量补回，计划下一轮。如此反复，直到所有人疏散完毕。另一个是利用蚁群算法设计逃生路径。将一定数量的蚂蚁分散在所有的受灾点，并且每一只蚂蚁身上携带有相同信息素，每经过一段弧就在该弧留下信息素。基于蚂蚁的行进过程具有时间周期性，每过一个周期，先判断弧上的蚂蚁是否到达终点，其次对于未到终点的蚂蚁，根据邻近弧上的信息素和预计的邻近节点到终点的最短距离来选择下一个行进方向，完成选择后更新每条弧上的信息素，不断循环直到所有蚂蚁都到达终点。

本文以同济大学南楼为例，结合模型进行逃生模拟。假设发生灾难，并且此灾难不会对逃生路径造成影响，将教室视为受灾点，假设每个人都从教室逃出且忽略每个人到达教室门口的时间。利用动态网络流的算法，解出了两组各个受灾点内人数随机的情况下，每个受灾点所有人的撤离时间以及南楼所有人的撤离时间。利用蚁群算法，解出每个人的撤离路径及所有人的撤离时间。如果遇到其它会对逃生路径造成影响的灾难，例如火灾烟雾会影响起火点附近的路，依然可用此模型，但需要根据实际情况修改图，将某些弧段删除，或是弧段随着时间的变化而减少。

关键词: 快速撤离问题　　图论模型

1 问题重述

1.1 背景

随着城市化进程加速，大城市的建筑密集程度和人口密度迅速增加，一旦发生突发事件，例如地震和火灾，由于建筑结构的复杂和建筑规模的庞大，其人员快速疏散策略显得尤为重要。

1.2 问题一

选取同济大学的一个熟悉的建筑，假设其发生地震或火灾，构想一个灾难场景，给出使建筑内的人快速撤离出来的方案，求出撤离所需的最短时间。

1.3 问题二

不同地方发生着不同的灾难，带来了不同的后果，找到这些不同中的相同点，结合问题一，给身处灾难的人提供逃跑准则，建立适用于大多数灾难场景的逃跑准则数学模型。

1.4 问题三

将建立的“逃跑准则”数学模型写成一篇告示，以便宣传和普及灾难发生时的逃跑准则。

2 问题分析

2.1 “均匀灾难”下的逃生模型

对于一个分布有人员的建筑物，并且每个人都能沿着建筑物的某条或者某几条路到达出口。如果发生一场灾难，并且这场灾难对每个人的可选逃生路径都没有影响，或者影响相同，可称其为“均匀灾难”，例如强度中等的地震。希望能够使建筑物内的人尽快逃生，这里尽快逃生包含的含义有: 所有人完全撤离的时间最短，并且在这个前提下，每个人逃生时间的总和达到最短。

那么每个人都能任意从这些逃生路径中选择，他的逃生时间受到路径长短，逃生速度，以及是否有人员拥堵而产生等待时间的影响。

对于人员较多的情况下，考虑个体行为，很难预测疏散的时间。于是可以将人员的流动看作是水流，人员在路径上的密度，会影响其速度。那么可以求出一个密度，使得单位时间的人流量达到最大。

然后本文利用图论模型模型解决问题，将这个建筑物看作一个无向图 $G(V,E)$，节点 V 包含受灾点，中间节点，出口，以及一个与所有出口相连的超级终点；E 是连接这些节点的弧。人员从受灾点出发到达超级终点。根据建筑实际的情况，弧所代表的路段应有长度以及一个单位时间最大人流量的信息。根据这些信息，安排路径，使得每秒钟每个弧上的人流都不超过其最大人流量，然后以最终撤离时间作为目标函数，在弧的限制条件下达到最小。

2.2 “非均匀灾难”下的逃生模型

不同地方发生不同灾难，我们同样可以建立上述图论模型进行求解最优逃生方案，需要做出改变的是，在发生“非均匀灾难”时，由于灾难对每条逃生路径的影响不同，并非所有的路径、出口都是可用于逃生的，因此需要将这些路径、出口从图论模型中删除，完善模型后用 2.1 的求解方法可解决此问题。

2.3 “逃跑准则”的建立

在不同地方发生不同灾难的情形上，通过大量模拟求解最优的逃生方案，总结其中的规律，以此作为“逃跑准则”。

3 模型假设

1. 假设每个人的心理素质和身体素质一样，均具有足够的身体条件跑到安全地点，不具有差异性。
2. 假设每个人都以同样的速度逃生，忽略在人流极少的情况下速度的变快。
3. 假设发生灾害时，建筑内的人可以及时且同时被告知危险，并立刻开始撤离。
4. 假设每个出口都是完全可用的。
5. 假设疏散人员是清醒状态，在疏散开始的时刻同时井然有序地进行疏散，且在疏散过程中不会出现中途返回选择其他疏散路径。
6. 受灾人员只要逃出出口即视作安全。

4 符号说明

符号	意义
$G(V,E)$	待疏散网络
V	节点集，v_i 为节点集中的元素
E	弧集，其中 e_{ij} 表示连接节点 i,j 之间的弧（$i,j \in V$）
l_{ij}	弧 e_{ij} 的长度
t_{ij}	通过弧 e_{ij} 的旅行时间
c_{ij}	弧 e_{ij} 的单位时间内允许通过的最大人流量
SE	超级终点，与所有出口相连接
S	受灾点的集合
W	表示受灾点
q^w	第 w 个受灾点待疏散人员总人数
P_m^w	m 撤离轮次第 w 个受灾点到超级终点 SE 所采用的路径集合
p_k^w	第 k 条路径（从第 w 个受灾点出发）
$f_k(t)$	t 时刻由路径 p_k^w 到达 SE 的动态流量
δ_{ijk}^w	$0-1$ 变量，如果 $e_{ij} \in p_k^w$，则 $\delta_{ijk}^w = 1$，否则为 0
$T_{p_k^w}$	第 w 个受灾点待疏散人员沿着路径 p_k^w 到达 SE 所需要的旅行时间，显然有 $T_{p_k^w} = \sum_{i,j} t_{ij}\delta_{ijk}^w$

符号	意义
T^w	第 w 个受灾点实际疏散结束时间
T	整个疏散网络疏散结束时间，即最后一个人离开危险区的时间，$T = \max_w\{T^w\}$
C_{ij}	弧 e_{ij} 的最大容量
$b_{ij}(t)$	t 时刻在 e_{ij} 弧上的人的数量
$\tau_{ij}(t)$	t 时刻在 e_{ij} 弧上所有的信息素
η_{ij}	节点 i 到 j 的先验概率
$m(t)$	t 时刻仍未到达 SE 的人数，显然有 $m(t) = \sum_{i,j} b_{ij}(t)$
m_i	节点 i 上的人的数量
h	人的编号，$h = 1, 2, ..., m$
$tabu_h$	禁忌表，用于记录人 h 所走过的节点
H_{ij}^h	人 h 在 e_{ij} 弧上时对下一个节点的转移概率
α, β	分别是转移概率中的信息素和能见度的加权值
k_j	表示距离第 j 个位置最近的安全通道
d_{jk_j}	表示第 j 个位置到达安全通道的最短距离

5 模型建立与求解

5.1 模型的准备

5.1.1 疏散网络的建立

以同济大学南楼为例，如图 1，对其建立 $G(V, E)$ 疏散网络。由于南楼具有左右对称性，因此我们仅对教学楼左模型进行分析，如图 2 所示。

假设走廊两侧正对的两个教室可以视为一个整体，进而我们将其合为一个节点表示出，并且假设灾难发生时，人员都分布在教室里，这些节点作为受灾点集合 S, 每个元素可表示为 W，包含信息待疏散人数 q^w；在正对的两个教室之间设置一个节点，作为走廊节点，同时在靠近楼梯处，设置一节点，以连接大教室、楼梯口、小教室，除此之外还有出口，这些为中间节点；为了方便算法使用，设置了超级终点 SE，连接所有出口。

而弧集 E 则是连通各个节点的路段，其元素表示为 e_{ij}，包含信息: 弧的长度 l_{ij}，通过弧的旅行时间 t_{ij} 以及一段弧单位时间内允许通过的最大人流量 c_{ij}。其中弧的长度都为模型中的实际长度。教室门口到走廊节点的弧以及出口到超级终点 SE 的弧在实际情形中长度 l_{ij} 为零，但 c_{ij} 不为零。

5.1.2 最佳人流密度和速度的确定

我们假设所有人在所有时刻的行走速度 v 保持一致，人流密度 ρ 也保持一致，一条路的宽度均匀，设为 d。对于人流量 c_{ij} 有以下公式

$$c_{ij} = vd\rho \tag{1}$$

其中 v 会收到 ρ 以及其他安全系数的影响，根据文献 [4]，在水平通道最佳的人流密度 $c_{ij} = 1.6$ 人 $/m^2$，$v = 1.8m/s$；楼梯上最佳的人流密度 $c_{ij} = 2.5$ 人 $/m^2$, $v = 1.3m/s$.

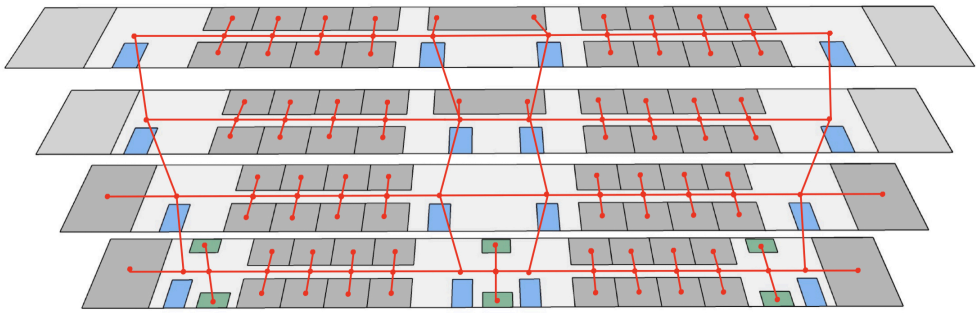

图 1: 疏散网络

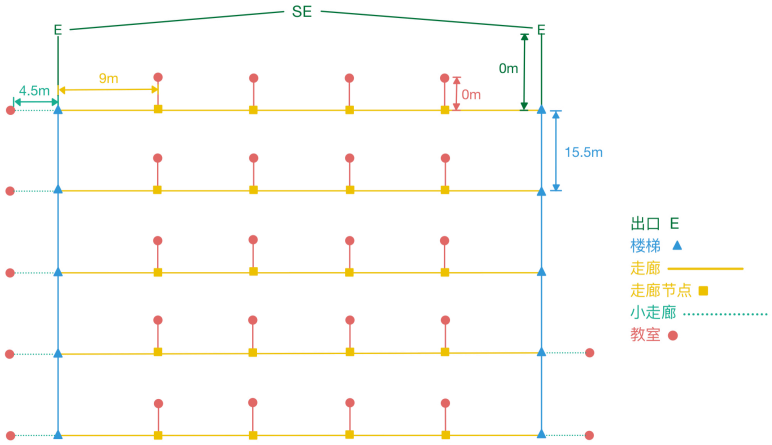

图 2: 教学楼左模型

5.2 模型的建立

该问题考虑的是如何对每一个受灾点待疏散人员进行合理分组，选择合适的路径进行疏散，使得疏散时间最小化。

对于每个受灾点 W 的疏散，可能会采取多条路径，可记为 p_k^w（这里 k 是对所有受灾点的撤离路径进行统一编号），它们组成了集合 P^w。对于每条路径，用 $0-1$ 变量 δ_{ijk}^w 记录弧 e_{ij} 是否在路径之中，用 $T_{p_k^w}$ 表示第 w 个受灾点待疏散人员沿着路径 p_k^w 到达 D_0 所需要的旅行时间 [2]，显然有

$$T_{p_k^w} = \sum_{i,j} t_{ij} \delta_{ijk}^w \tag{2}$$

而每条路径还对应了一个动态的流量 $f_k(t)$，其含义是 t 时刻单位时间沿路径 p_k^w 从超级终点 D_0 撤离出的人数，则 T 时刻沿该条路径撤离的总人数为

$$x_k^w(T) = \sum_{t=0}^{T} f_k^w(t) \tag{3}$$

对于受灾点 W，其实际疏散结束时间为 T^w，则有

$$q^w = \sum_{p_k^w \in P^w} x_k^w(T^w) = \sum_{p_k^w \in P^w} \sum_{t=0}^{T^w} f_k^w(t) \tag{4}$$

在此问题中我们须使总撤离时间 $T = \max\{T^w\}$ 最小，并且还要满足每个时刻每条弧段上的人流量都是小于它的容量的，所以可以建立模型

$$\begin{aligned} & \min T = \max_w T^w \\ s.t. \quad & \sum_{p_k^w} \delta_{ijk}^w f_k^w(t + T_{ijk}^w) \le c_{ij}, \quad \forall (i,j) \in E, t = 1,2,...,T \\ & q^w = \sum_{p_k^w \in P^w} x_k^w(T^w) = \sum_{p_k^w \in P^w} \sum_{t=0}^{T^w} f_k^w(t) \\ & f_k^w(t) \ge 0 \end{aligned} \tag{5}$$

其中 T_{ijk}^w 表示 p_k^w 路径中从 e_{ij} 到超级终点的旅行时间。

5.3 基于动态网络流的模型求解

对于该模型不可能遍历所有方案求得最短疏散时间，所以利用动态网络流来设计算法。

对于待疏散网络 $G(V,E)$，我们动态地考虑疏散过程，进行一轮一轮的疏散，尽量饱和疏散最短路径。

疏散网络有多个受灾点，所有出口都与超级终点 SE 连接，要使受灾点的人都往超级终点撤离。根据实际生活的经验，一个人逃生，一般会选择最短路径逃生，但如果路上人很多可能会因为拥堵而出现停滞，此时选择其他的路径可能还会缩短逃生时间，所以要综合考虑所有人来设计方案。

$G(V,E)$ 图里有很多条连接受灾点与超级终点的路径，为使疏散状态达到每时刻的最优状态，先将最短路径纳入第一轮疏散的计划内，设其为 p_k^w，使其动态流量在这一轮达到最大，记 f_k 为路径 p_k^w 的最大通行人流量，当计划完这条路径后，更新 $G(V,E)$ 中的弧段和弧段容量，再继续找最短路径，直到 $G(V,E)$ 中没有连通超级终点和受灾点的路径，由此规划完第一轮疏散的路径。在考虑第一轮疏散时，先不考虑那些没有纳入第一轮疏散计划的受灾点。对于涉及到的受灾点，若受灾点在此轮有多条计划的疏散路径，则按照到达时间相同原则来规划人员的分配（为使计划路径可满足此原则，需在纳入路径时就进行一个判断）。

然后进行第一轮的疏散，只要有一个受灾点全部撤离后，则视为一轮疏散结束，将时间记为 T_1，更新剩余受灾人数，对于空出来的路径和多出来的容量进行下一轮计划。在考虑第二轮计划的待疏散人群时，我们不好直接预计 T_1 时，它们具体在那个位置，可以从 $t = 0$ 时开始模拟，中途可能会有因为第一轮的人群没有疏散完而停滞的情况，在这里，无论它在哪个位置有停滞的情况，都满足 T_1 之前，撤出人数为零。据此可以算出第二轮的各个受灾点的完全撤离时间，然后确定 T_2。

以下是准备定理和公式以及具体的算法步骤。

5.3.1 准备定理和公式

1. 路径 p_k^w 的最大通行容量为

$$f_k = \min\{c_{ij} | e_{ij} \in p_k^w\} \tag{6}$$

2. 第 w 个受灾点的人沿着路径 p_k^w 到达 SE 所需要的旅行时间为 $T_{p_k^w} = \sum_{i,j} t_{i,j} \delta_{ijk}^w$

3. 若 p_k^w 为第 m 轮疏散的某一条路径，前 $m-1$ 轮撤离所花总时长为 T，则在 t 时刻经由路径 p_k^w，到达超级终点 SE 的总人数 $x_k^w(t,T)$ 满足：

$$x_k^w(t,T)=\begin{cases}(t-(\max\{T_{p_k^w},T\}-1))f_k, & t\geq\max\{T_{p_k^w},T\}\\ 0, & t<\max\{T_{p_k^w},T\}\end{cases} \tag{7}$$

4. 对于一个第 m 轮次在疏散的受灾点 W，前 $m-1$ 轮撤离所花总时长为 T，其待疏散人数为 q^w，若已选中 $p_1,p_2,...,p_y$ 作为该疏散轮次疏散该受灾点的路径，(这些路径满足 $q^w\geq\sum_{k=1}^{y}x_k^w(T',T)$ $(T'=\max\{T_{p_k}|k=1,...,y\})$，则当所有用于疏散的路径最后一人到达超级终点 SE 时刻相等时，疏散该受灾点所花时间最短，即

$$q^w=\sum_{k=1}^{y}x_k^w(T_1+T,T) \tag{8}$$

其中 T_1 为 W 在此轮疏散所花时间。

5.3.2 算法实现步骤

1. 初始化：输入疏散网络 $G(V,E)$，每条弧的初始容量 c_{ij} 及旅行时间 t_{ij}，受灾点集合 S，各个受灾点的待疏散人数 q^w，增加 SE 为超级终点，连接各个出口 r.

   m 撤离轮次的撤离受灾点集合为 $S_m=\emptyset$，m 轮次从第 w 个受灾点出发的路径集合为 $P_m^w=\emptyset$，m 轮次各个受灾点完全撤离时间集合 $TP_m=\emptyset$，m 轮次路径不可加受灾点集合为 $\widehat{S_m}=\emptyset$，m 轮次受灾点内无人，但撤离路径上有人的受灾点集合为 S_m^o.

   令撤离轮次 $m=1$，路径编号 $k=1$，m 轮次第 w 个受灾点完全撤离时间 $T_m^w=0$（$w=1,2,...,s$），总撤离时间 $T=0$。

2. 判断 S 中是否存在受灾点，存在，则进入步骤 3，不存在，跳至步骤 11.

3. 利用 Dijkstra 算法 [3] 求出 $S-\widehat{S_m}$ 中各个受灾点到超级终点的最短路径（旅行时间最短），记为 p^w，这些路径组成集合 P_k，再从 P_k 中选出 T_{p^w} 最小的一条路径，记为 p_k^w，其对应的受灾点为 W.

4. 计算路径 p_k^w 的最大通行容量 f_k.

   若 $q^w<\sum_{p_j^w\in P_m^w\bigcup\{p_k^w\}}x_j^w(T_{p_k^w},T)$（这意味着如果新增这条路径并不能使该受灾点的疏散时间得到减少），则令 $\widehat{S_m}=\widehat{S_m}\bigcup\{W\}$，并跳至第 3 步，否则跳至第 5 步.

5. 令 $P_m^w=P_m^w\bigcup\{p_k^w\}$，$S_m=S_m\bigcup\{W\}$

6. 更新每条弧上的最大人流量，$c_{ij}=\begin{cases}c_{ij}-f_k & e_{ij}\in p_k^w\\ c_{ij}, & e_{ij}\notin p_k^w\end{cases}$，如果 $c_{ij}=0$，则从原有网络中删除弧 e_{ij}，更新网络.

7. 如果更新的网络不存在任意一条从 S 中的受灾点到超级终点的路径，则转到步骤 8，否则，令 $k=k+1$，转到步骤 3.

8. 此时确定了 m 疏散轮次的疏散路径，对于 S_m 中的受灾点，依次更新 T_m^w.

   (a) 若 $W\in S_m^o$，则 T_m^w 保持不变.

(b) 若 $W \notin S_m^o$，至少要等到 T 后才会有人撤出，要使所花时间最短，则有

$$q^w = \sum_{p_k^w \in P_m^w} x_k^w (T_m^w + T, T) \tag{9}$$

令 $TP_m = TP_m \bigcup \{T_m^w\}$.

$T_m = \min TP_m$.

9. 令 $S_{m+1} = S_m$，对于 S 中的每一个受灾点有 $P_{m+1}^w = P_m^w$

对于 S_m 中的每一个受灾点 W，更新 $q^w = q^w - \sum_{p_k^w \in P_m^w} x_k^w (T_m + T, T)$，

若 $q^w < \sum_{p_k^w \in P_m^w} x_k^w (T', 0)$（$T' = \max\{T_{pk}{}^w | p_k^w \in P_m^w\}$），则从原有网络 S 中删除受灾点 W，并记 $T_{m+1}^w = T_m^w - T_m$.

(a) 若 $T_{m+1}^w \neq 0$，则令 $TP_{m+1} = TP_{m+1} \bigcup \{T_{m+1}^w\}$，$S_{m+1}^o = S_{m+1}^o \bigcup \{W\}$.

(b) 若 $T_{m+1}^w = 0$，则从 S_{m+1} 中删除受灾点 W，并将 W 对应的路径集 P_m^w 中占用的弧段和容量补回，$E = E \cup \{e_{ij}\}$（$e_{ij} \in p_k^w$），设新补回的弧段容量为 0.

更新每条弧的最大容量 $c_{ij} = c_{ij} + f_k \quad e_{ij} \in p_k^w$.

10. $T = T + T_m$，令 $m = m + 1$，跳回步骤 2.

11. 若 $TP_m = \emptyset$，则结束此算法，若其不为空，令 $T = T + \max TP_m$.

5.4 基于蚁群算法的模型求解

该模型还可用改良蚁群算法来进行疏散路径的设计。在系统初始时刻，每一条路径中包含的信息量是完全相等的，对于任何人可以随机选择每条路径。而蚁群算法是根据有向图的知识，寻找最优的路径。人 h（$h = 1, 2, ..., m$）在其运动的过程中，根据当前系统中每一条安全路径所含有的信息量，进行下一步的方向选择。同时，使用禁忌表 $tabu_h$（$h = 1, 2, ..., m$），记录人所走过的节点，从而可根据禁忌表进行疏散路径的动态调整。在疏散过程中，其状态转移概率可以通过各路径的信息量及启发信息计算，其转移概率 [1] 为：

$$P_{ij}^h = \begin{cases} \dfrac{\tau_{ij}^\alpha(t)\eta_{ij}^\beta(t)}{\sum \tau_{ij}^\alpha(t)\eta_{ij}^\beta(t)}, & j \in \{V - tabu_h\} \\ 0 & else \end{cases} \tag{10}$$

当 $\alpha = 0$ 时，转移概率的计算中忽视了之前的疏散经验，从而转换为贪心疏散；当 $\beta = 0$ 时，下一个节点的能见度为 0. 随着时间的变化，需要对残余的信息素进行更新，其中，参数 ρ（$0 \leq \rho \leq 1$）是信息素的挥发率；$(1 - \rho)$ 表示信息素的残留率，其体现了信息素的浓度的持久性，各路段上信息素的更新可表示为：

$$\tau_{ij}(t+1) = (1 - \rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{11}$$

$$\Delta\tau_{ij}(t) = \sum_{h=1}^{m} \Delta\tau_{ij}^h(t) \tag{12}$$

$$\Delta\tau_{ij}(t) = \frac{1}{h\text{已走过的路径长度}} \tag{13}$$

而启发式函数 [1] 为

$$\eta_{ij} = \begin{cases} \dfrac{1}{l_{ij} + \vartheta_k d_{jk_j}} \\ \dfrac{}{\sum\limits_{j=1}^{m_i} (\dfrac{1}{l_{ij} + \vartheta_k d_{jk_j}})} \end{cases} \tag{14}$$

$\vartheta_k = M/(P_{e_k})$ 中，P_{e_k} 表示第 k 个安全通道可以容纳逃生人员的数量；M 为所有的安全通道总共可以容纳的人员数目。

在每次路径搜索开始，首先使用较大的 ρ 值，增加其收敛速率。当搜索到最优路径的某一个范围时，适当的减小 ρ 值，从而增加其搜索的精度。

5.4.1 符号准备

1. judgepoint：judgepoint(i) 为中储存了在节点 i 待判断走向的蚂蚁。
2. antfollow：跟踪了每只蚂蚁的起始节点，目标节点和剩余旅行时间。

5.4.2 算法实现步骤

1. 初始化，蚂蚁的初始节点为其所在的教室节点，假设教室人数一样，为 num。初始化信息素矩阵，信息素初值为常数；初始化信息素挥发率 $\rho = 0.5$；计算启发式函数 $\eta_{ij} =$，初始化 $\alpha = 6$，$\beta = 4$。
2. 时间 t 增加 T_0（其中 T_0 小于等于蚂蚁区域发生变化的最短时间）。更新 b_{ij} 中蚂蚁跟踪数据中的剩余旅行时间 t_s，判断 t_s 是否为零，若是，则从 b_{ij} 中删除此蚂蚁，又若 j 是安全通道，建筑内蚂蚁数 $m(t)$ 减 1，若不是，则加入 judgepoint(j) 中。
3. 若 S 中有蚂蚁走到对应走廊节点上 k 上，则加入 judgepoint(k) 中。
4. judgepoint(i) 中的蚂蚁 h，与 v_i 相连的节点中去除教室节点与禁忌表 $tabu_h$ 中的节点，更新蚂蚁 h 的跟踪数据，将新选择的节点加入到禁忌表 $tabu_h$（$h = 1, 2, ..., m$）中。
5. 更新信息素 τ_{ij}。
6. 判断蚂蚁是否全都到达终点，即 $m(t)$ 是否为零，若不是，则返回步骤 2；若是，则结束本次模拟。

6 结果分析与检验

6.1 “均匀灾难”求解

对于“均匀灾难”的求解，我们以南楼为例，假设发生“均匀灾难”。

6.1.1 模型数据确定

如图 2，对于每段弧有如表 1 的数据

注：每个教室延伸出的弧的宽度是教室门的宽度，对于大教室和中间教室，其节点由两个门合成，故宽度为实际宽度的两倍；由出口延伸的弧的宽度也为门的宽度的两倍。

弧的种类	大教室到楼梯节点	中间教室到走廊节点	右侧教室到大厅节点	大厅节点到出口	楼梯节点到出口
弧的长度 (m)	4.3	0	4.3	8.2	4.6
弧的宽度 (m)	2.8	2.08	1.04	3.6	3.08
人的通行速度 (m/s)	1.8	1.8	1.8	1.8	1.8
旅行时间 (s)	2.39	0	2.39	4.56	2.56
最大人流量 (人 $/s$)	8.06	6.00	3.00	10.37	8.87

表 1: 弧的信息（1）

弧的种类 (m)	走廊节点到走廊（或楼梯）节点	楼梯节点到楼梯节点	大厅节点到大厅节点	走廊节点到大厅节点
弧的长度 (m)	9	15.55	15.55	6.9
弧的宽度 (m)	2.3	2	2	2.3
人的通行速度 (m/s)	1.8	1.3	1.3	1.8
旅行时间 (s)	5	11.96	11.96	3.83
最大人流量 (人 $/s$)	6.62	6.5	6.5	7.48

表 2: 弧的信息（2）

6.1.2 利用动态网络流对模型的求解

对于每一个受灾点（教室）的受灾人数，根据受灾点的大小，我们取不超过 150 的随机数，对于不同的人群分布，利用动态网络流算法，取了两组数据，有如表 3，表 4 的结果。其中每个受灾点的标号如图 3 所示。

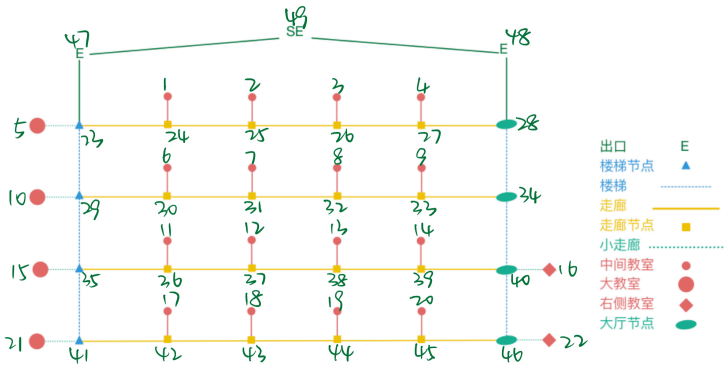

图 3: 动态网络流解法对每个受灾点的标号

根据此模型，对于一个四层楼，且受灾点较稠密分布的建筑，完全疏散 1000 多个人要花 100 多秒的时间，如果发生的灾难有足够的逃生时间，如轻微的地震，那么可以所有人按照这个模型逃生。但如果可逃生时间不足，有一部分人应选取较为安全的地带躲避。例如，当发生较为严重的地震时，离出口较远的人应就近找坚固的遮蔽物进行躲避，尽量靠近水源。

6.1.3 利用蚁群算法对模型的求解

依然取 6.12 里的两组受灾人数的数据，利用蚁群算法，得出了如下结果。图 4 为该算法对每个节点的标号。

图 5 显示了当 $\alpha = 0.1$，$\rho_0 = 0.9$ 时，撤离时间关于 β 的变化，当 $\beta = 3$ 时，撤离时长最短。图 6 显示了当 $\beta = 3$，$\rho_0 = 0.9$ 时，撤离时间关于 α 的变化。当 $\alpha = 6$ 时，撤离时长最短。图 7 显示了当 $\alpha = 6$，$\beta = 3$ 时，撤离时长关于信息素挥发速率 ρ_0 的变化，当 $\rho_0 = 0.1$ 或 0.5 时，撤离时

受灾点编号	1	2	3	4	5	6	7	8	9	10	11	12
受灾点人数	57	85	11	8	79	116	140	19	85	70	1	50
受灾点撤离总时长	21.28	29.56	30.51	29.89	13.75	42.01	52.41	34.00	32.20	28.68	47.85	54.18
受灾点编号	13	14	15	16	17	18	19	20	21	22	总	
受灾点人数	24	119	46	79	24	90	39	98	103	112	1455	
受灾点撤离总时长	70.81	74.54	48.68	69.33	73.48	86.17	90.45	98.62	68.81	106.81	106.81	

表 3: 受灾点受灾人数与疏散时间（模拟 1）

受灾点编号	1	2	3	4	5	6	7	8	9	10	11	12
受灾点人数	27	39	21	20	130	86	82	21	127	93	52	76
受灾点撤离总时长	19.75	22.89	15.89	10.72	18.08	43.21	52.54	43.38	42.47	30.88	62.54	63.66
受灾点编号	13	14	15	16	17	18	19	20	21	22	总	
受灾点人数	60	11	35	18	27	35	62	7	135	141	1305	
受灾点撤离总时长	54.21	46.04	54.88	47.38	82.14	84.98	80.37	64.66	79.64	102.58	102.58	

表 4: 受灾点受灾人数与疏散时间（模拟 2）

长较短。实际上，让信息素挥发率随时间变化可以使它收敛更快 [1]。而由图 8 可以看出由图可以看出，当教室人数较少时，撤离时长较长，因为蚂蚁少，留下的信息素少，对于之前蚂蚁的“经验”没有很好的接收到；当教室人数较多时，撤离时间也较长，因为蚂蚁会拥堵，因此绕路。

6.2 “非均匀灾难”求解

对于“非均匀灾难”，灾难对每条路径和节点的影响程度不同，依然可以用图论模型，但需要改变图的节点，弧以及它们的相关数据，可以求出完全疏散所有人的时间。对于一些可能设置有紧急避难的地点的建筑，在图 $G(V, E)$ 中就可以设几个有容纳上限的安全出口与之对应。

6.3 “逃生准则”

1. 发生灾难时首先判断安全逃生时间和撤离所需时间，如果时间够，则迅速撤离，如果时间不够，则找紧急避难点躲避。
2. 寻找安全通道标志，沿着指向走。
3. 判断每个逃生路径的人流量，避开拥堵路径。
4. 如果建筑内有疏散人员，听从指挥。
5. 不要拥挤，谨防踩踏事件。

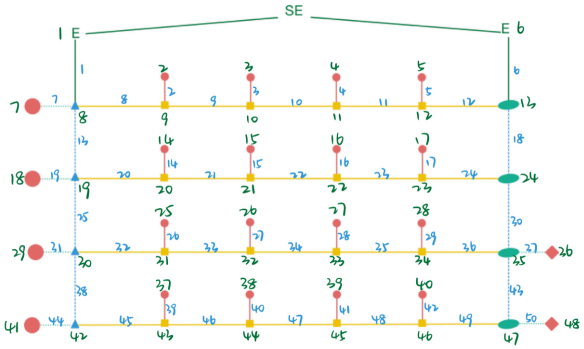

图 4: 蚁群算法中对每个节点的标号

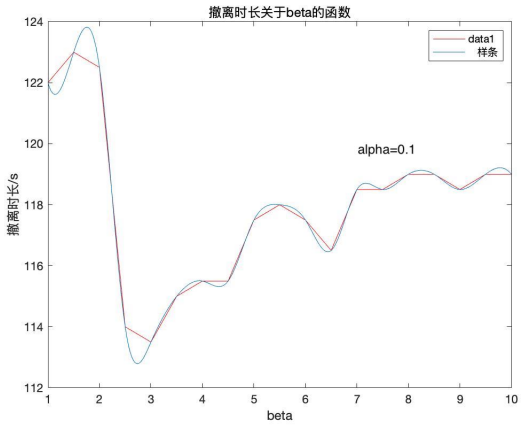

图 5: 撤离时长关于 β 的函数

7 模型评价与推广

7.1 模型评价

7.1.1 优点

本文采用图论模型解决快速撤离问题，其优点在于能够清晰的刻画出受灾点、逃生路径及逃生出口它们之间的量化关系。在不同地方发生不同的灾难时，仅仅需要对这栋建筑物所表示的邻接矩阵做些处理，便可以利用本文所给出的动态网络流或者蚁群算法得到相对最佳的逃生方案，与此同时，这两个算法的时间复杂度并不高，能够在极短时间内得出结果。

7.1.2 缺点

1. 我们将所有逃生者视为不具差异性的个体，但是在实际中，人与人之间具有一定的差异性。
2. 本文在计算最优路径时，考虑到人流密度与疏散速度的关系，为保证时间最小化，默认他们从教室出来后，按照最佳的速度，最恰当的人流密度逃生。但是在实际中，由于恐慌等心理因素，每个个体都想着以最快的速度逃生，这会产生拥挤现象，反而使得逃生的效率下降。
3. 实际上，从灾难发生到逃生的过程中，由于每个人的反应时间差异，需要一定的时间才能使得所有人都被告知危险，在这个模型当中是直接忽略了这部分时间。

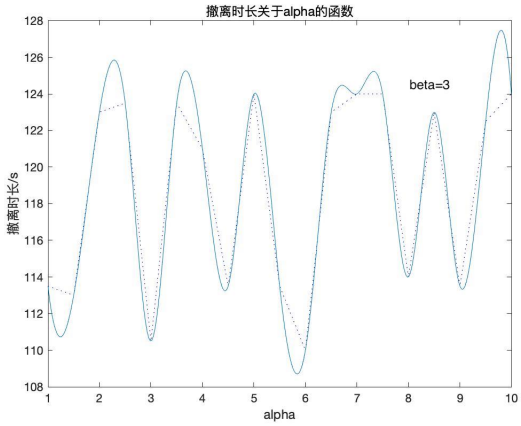

图 6: 撤离时长关于 α 的函数

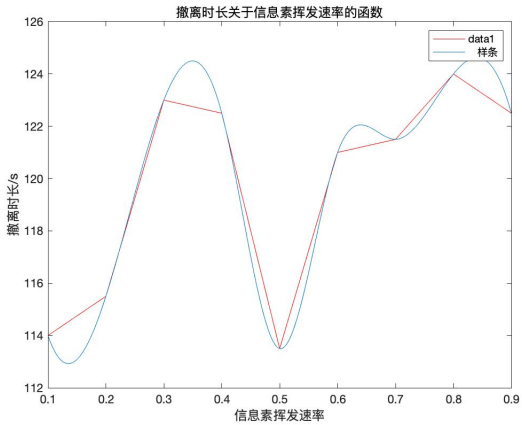

图 7: 撤离时长关于信息素挥发速率的函数

7.2 模型推广

本文所用的图论模型不仅仅能够解决发生灾难时的快速撤离问题，在查找文献的过程中，发现大部分用于解决 TSP 问题，因此该模型可以推广到寻找到 n 个城市的“最短”路径，这里由于此类问题至今没有找到一个有效算法，或者说更倾向于 NPC 问题不存在有效算法这一猜想，因此无法得到真正意义上的最短路径。

参考文献

[1] 何静，刘红霞，徐明霞. 基于改进蚁群算法的大型建筑物疏散路径设计 [A]. 陕西国防工业职业技术学院,2020:30-37.

[2] 高岩，王宏杰，杨建芳. 多层建筑物应急疏散模型和算法 [A]. 系统仿真学报,2014,02:267-273.

[3] 司守奎. 数学建模算法与应用 [M]. 北京，国防工业出版社，2011.08：40-41.

[4] 张云明. 安全疏散基础参数的量化 [A]. 中国人民武装警察部队学院,2011:496-499.

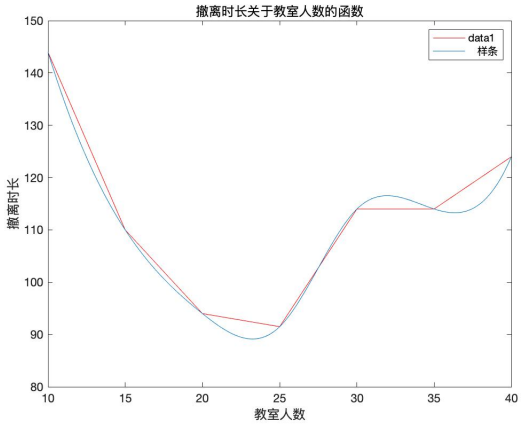

图 8: 撤离时长关于教室人数的函数

附录

```
1
2  function T = evacuation(A,V,c,w)
3
4  %   A表示时间的邻接矩阵; B表示容量矩阵;c表示受灾人数向量;前w个节点表示受灾点
5  %   矩阵的最后一个节点表示超级终点
6  n = size(A,1);
7  A0 = A; %记录最初始的邻接矩阵
8  p = zeros(1,n+2);    %存储路径, 其中前n项为节点, n+1项为逃生时间, n+2项为该路径最大容量
9  P = p;
10 S = zeros(4,w); %第一行表示m轮受灾点是否有人;第二行表示m轮是否安排受灾点疏散;
11 %第三行表示m轮待撤离人数; 第四行放m轮撤离时间
12 S(1,:) = ones(1,w);
13 S(3,:) = c;
14 s = ones(1,n);  %表示不可加受灾点,0表示不可加
15 T = 0;  %撤离总时间
16 S(1,:) = ones(1,w);
17 t = 0;  %标记路径时间
18 while sum(S(1,:)) ~= 0  %判断是否有受灾点
19     while t ~= inf  %是否有可选路径
20             [t,p] = find_optimalR(A,S(1,:),s,V);%贪心算法算出最短路径, 输出路径向量和时间
21             w1 = p(1);%记录第几个受灾点
22             if t ~= inf
23                 m = size(P,1);
```

```
                p_nop = calculate_nop(p,p(n+1),T);  % p路径下p(n+1)
    时刻逃生的人数
                for i = 1:m
                    p_nop = p_nop + calculate_nop(P(i,:),P(i,n+1),T)
    ; %计算所有路径逃生人数
                end

                if S(3,w1) < p_nop      %第4步的判断
                    s(w1) = 0;     %该轮次路径不可加
                else
                    P = [P;p];  %路径集合
                    S(2,w1) = 1;     %更新第m轮安排的受灾点
                    [A,V] = update_delete(A,V,p);%更新邻接矩阵
                end
            end
    end
    s = ones(1,w);
    t = 0;
    %此时，安排完第m轮
    for k = 1:w
        if S(2,k) == 1
            q_w = S(3,k);
            S(4,k) = calculate_T_w(P,T,q_w,k);%只算这一轮有计划的受
    灾点
        end
    end
    a = 1./zeros(1,w);
    for i = 1:w
        if S(4,i) ~= 0
            a(i) = S(4,i);
        end
    end
    T1 = min(a);
    m = size(P,1);
    for k = 1:w
        if S(2,k) == 1
            q1 = 0;
            for i = 1:m
                if P(i,1) == k
                    q1 = q1 + calculate_nop(P(i,:),T1+T,T);
                end
            end
            S(3,k) = S(3,k) - q1;
```

```
64          end
65      end
66      %更新待撤离人数(这里只更新这一轮有计划的)
67      for k = 1:w
68          if S(2,k) == 1 || S(4,k) ~= 0
69              S(4,k) = S(4,k) - T1;
70              m = size(P,1);
71              q2 = 0;
72              a = zeros(1);
73              for i = 1:m
74                  if P(i,1) == k
75                      a = [a  P(i,n+1)];
76                  end
77              end
78              T2 = max(a);
79              for i = 1:m
80                  if P(i,1) == k
81                      q2 = q2 + calculate_nop(P(i,:),T2,0);
82                  end
83              end
84              if S(3,k) < q2
85                  S(1,k) = 0;
86                  S(2,k) = 0;
87              end
88              if S(4,k) == 0
89                  [A,V,P] = update_add(A,A0,V,P,k);
90              end
91          end
92      end
93      T=T+T1;
94  end
95  T=T+max(S(4,:));
96
```

```
1  function f_k = calculate_minv (V,p)
2  %    function f_k = calculate_minv (V,R)
3  %    V为存储各点之间的最大容量，p为某个路径
4  %    返回的f_k为该路径的最大饱和容量
5
6  n = size(V,1);
7  f_k = inf;  %设置默认容量为无穷，方便求饱和容量
8  for i = 1:n
```

```
    if p(i) ~= 0 && p(i+1) ~= 0
        f_k = min( f_k,V(p(i),p(i+1)) );
    end
end
p(n + 2) = f_k;
```

```
function x = calculate_nop(p,t,T)

%   function x = calculate_nop(p,t,T),calculate_numbers of people
%   p 为某条路径, t 为当前时刻, T为前m-1轮所花的总时长

len = size(p,2);
flag = max(p(len-1),T);
if t >= flag
    x = ( t-(flag - 1) )*p(len);
else
    x = 0;
end
```

```
function T_w = calculate_T_w(P,T,q_w,w)

%   function T_w = calculate_T_w(P,T,q_w)
%   给出多个路径向量并成的矩阵P,T为前m-1轮所花时间
%   q_w为w受灾点的人数

m = size(P,1);
n = size(P,2);
a = q_w;  %a为分子
b = 0;  %b为分母
for i = 1:m
    if P(i,1) == w
        a = a + (max(P(i,n-1),T) - 1)*P(i,n);
        b = b + P(i,n);
    end
end
T_w = a/b-T;
```

```
function T = evacuation_time(A,p)

%   function T = evacuation_time(A,p)
%   A 为时间的邻接矩阵, p为存储某条路径的向量, 返回沿该路径所花的时间
```

```
5
6  n = size(A,1);
7  T = 0;
8  for i = 1:n
9      if p(i) ~= 0 && p(i+1) ~= 0
10         T = T + A(p(i),p(i+1));
11     end
12 end
13 p(n + 1) = T;
```

```
1  function [T,p] = find_optimalR(A,S,S_m,V)
2
3  %   function [T,p] = find_optimalR(A,S,S_m)
4  %   S 为受灾点是否为空的向量，S_m为第m轮次后受灾点是否为空的向量，
5  %   A为邻接矩阵
6  %   查找出受灾点最优的路径
7
8  w = size(S,2);
9  F = zeros(1,w);
10 n = size(A,1);
11 S_f = zeros(1,w);    %经过m轮次后仍有人员的受灾点
12 for i = 1:w
13     if S(i) == 1
14         F(i) = 1;
15     end
16 end
17 for j = 1:w
18     if F(j) == 1 && S_m(j) == 1
19         S_f(j) = 1;
20     end
21 end
22 % 得出S，S_m中同时为1的受灾点
23 r = 1./zeros(1,w);
24 for j = 1:w
25     if S_f(j) == 1
26         r(j) = mydijkstra(A,j,n);
27     end
28 end
29 if min(r) == inf
30     T = inf;
31     p=[1  2]; %随机赋值路径，保证函数输出。
32 else
```

```
    [~ ,k] = min(r);     %k为最小值的下标，即我们所需的受灾点
    [T,p] = mydijkstra(A,k,n);
    temp = size(p,2);    %记录路径的节点数
    p(temp+1 : n+2) = 0;     %扩大R，用0填充为n+2维向量
    p(n + 1) = T;    %在n+1位置上赋值路径所需时间
    if temp > 0     %找到路径
        a = zeros(1,temp-1);     %用于存储节点间的最大容量
        for k = 1:temp-1
        a(k) = V(p(k),p(k+1));
        end
        p(n+2) = min(a);
    else
    p(n+2) = 0;
    end
end
```

```
function [mindistance,path] = mydijkstra(A,start,destination)

%  function [mindistance,mypath] = mydijkstra(A,start,destination);
%  输入：A邻接矩阵；A(i,j)i到j之间的距离。
%  sb起点，db终点。
%  mindistance最短路的距离，mypath最短路的路径

n = size(A,1);
visited(1:n) = 0;
distance(1:n) = inf;
distance(start) = 0; %起点到各顶点距离的初始化
visited(start) = 1;
u = start; % u为最新的P标号顶点
parent(1:n) = 0; % 前驱顶点的初始化
for i = 1:n-1
    id = find(visited == 0); %查找未标号的顶点
    for v = id
        if A(u,v) + distance(u) < distance(v)
            distance(v) = distance(u) + A(u,v); %修改标号值
            parent(v) = u;
        end
    end
    temp = distance;
    temp(visited == 1) = inf; % 已标号点的距离换成无穷
    [t,u] = min(temp); %找标号值最小的顶点
    visited(u) = 1; %标记已经标号的顶点
```

```
end
path = [];
if parent(destination) ~= 0 %如果存在路!
    t = destination; path = destination;
    while t ~=start
            p = parent(t);
            path = [p path];
            t = p;
    end
end
mindistance = distance(destination);
```

```
function [A,V,P] = update_add(A,A0,V,P,k)

%   function [A,V,R_mat]=update_add(A,A0,V,P,k)
%   更新邻接矩阵A，容量矩阵V，以及路线矩阵P
%   A0为最原始的邻接矩阵，k 为第k个受灾点

m = size(P,1);
n = size(P,2);
for i = 1:m
    if P(i,1) == k
        for j = 1:n-3
            if P(i,j) ~= 0 && P(i,j+1) ~= 0
                A(P(i,j),P(i,j+1)) = A0(P(i,j),P(i,j+1));
                A(P(i,j+1),P(i,j)) = A0(P(i,j+1),P(i,j));
                V(P(i,j),P(i,j+1)) = V(P(i,j),P(i,j+1)) + P(i,n);
                V(P(i,j+1),P(i,j)) = V(P(i,j+1),P(i,j)) + P(i,n);
            end
        end
    end
end
i = 1;
while i <= m  % 删除P矩阵中，已用过的路径
    if P(i,1) == k
        P(i,:) = [];
        m = m - 1;
    else
    i = i + 1;
    end
end
```

```
1
2 function [A,V] = update_delete(A,V,p)
3
4 %    function [A,V] = update(A,V,p)
5 %    根据路径p的信息更新邻接矩阵A, V
6
7 n = size(p,2);
8 for i = 1:n-3
9     if p(i) ~= 0 && p(i+1) ~= 0
10        V(p(i),p(i+1)) = V(p(i),p(i+1)) - p(n);
11        V(p(i+1),p(i)) = V(p(i+1),p(i)) - p(n);
12        if V(p(i),p(i+1)) == 0  %容量为0的路设为断路
13            A(p(i),p(i+1)) = inf;
14            A(p(i+1),p(i)) = inf;
15        end
16    end
17 end
```

```
1 function [t,e,antfollow2]=escape(num,Alpha,Beta,p0)
2 %快速撤离
3 %num_教室人数  Alpha_信息素影响因子 Beta_能见度加权值 p0_信息素挥发
      因子
4
5 %初始化
6 A=[[1,8,1,0,0,0,0,0,0];[1,9,2,0,0,0,0,0,0];[1,10,3,0,0,0,0,0,0];
7 [1,11,4,0,0,0,0,0,0];[1,12,5,0,0,0,0,0,0];[1,13,6,0,0,0,0,0,0];
8 [1,8,7,0,0,0,0,0,0];[4,1,1,7,7,9,8,19,13];[3,2,2,8,8,10,9,0,0];
9 [3,3,3,9,9,11,10,0,0];[3,4,4,10,10,12,11,0,0];
10 [3,5,5,11,11,13,12,0,0];[3,6,6,12,12,24,18,0,0];
11 [1,20,14,0,0,0,0,0,0];[1,21,15,0,0,0,0,0,0];
12 [1,22,16,0,0,0,0,0,0];[1,23,17,0,0,0,0,0,0];[1,19,19,0,0,0,0,0,0];
13 [4,8,13,18,19,20,20,30,25];[3,14,14,19,20,21,21,0,0];
14 [3,15,15,20,21,22,22,0,0];[3,16,16,21,22,23,23,0,0];
15 [3,17,17,22,23,24,24,0,0];[3,13,18,23,24,35,30,0,0];
16 [1,31,26,0,0,0,0,0,0];[1,32,27,0,0,0,0,0,0];
17 [1,33,28,0,0,0,0,0,0];[1,34,29,0,0,0,0,0,0];[1,30,31,0,0,0,0,0,0];
18 [4,19,25,29,31,31,32,42,38];[3,25,26,30,32,32,33,0,0];
19 [3,26,27,31,33,33,34,0,0];[3,27,28,32,34,34,35,0,0];
20 [3,28,29,33,35,35,36,0,0];[4,24,30,34,36,36,37,47,43];
21 [1,35,37,0,0,0,0,0,0];[1,43,39,0,0,0,0,0,0];[1,44,40,0,0,0,0,0,0];
22 [1,45,41,0,0,0,0,0,0];[1,46,42,0,0,0,0,0,0];[1,42,44,0,0,0,0,0,0];
23 [3,30,38,41,44,43,45,0,0];[3,37,39,42,45,44,46,0,0];
```

```
[3,38,40,43,46,45,47,0,0];[3,39,41,44,47,46,48,0,0];
[3,40,42,45,48,47,49,0,0];
[3,35,43,46,49,48,50,0,0];[1,47,50,0,0,0,0,0,0]];%邻接矩阵 连接的节点数k+(连接的节点+对应的弧序号)x k
w=[8,13];%安全通道编号
classroom=[2,3,4,5,7,14,15,16,17,18,25,26,27,28,29,36,37,38,39,40,41,48];%教室编号
bigroom=[7,18,29,36,41,48];%大教室编号
t=0;%初始化时间
edgelen=[0,0,0,0,0,0,4.5,9,9,9,9,9,15.5,0,0,0,0,15.5,4.5,9,9,9,9,9,15.5,...

0,0,0,0,15.5,4.5,9,9,9,9,9,4.5,15.5,0,0,0,0,15.5,4.5,9,9,9,9,9,4.5];%弧的长度
edgetime=[0,0,0,0,0,0,2.5,5,5,5,5,4,12,0,0,0,0,12,2.5,5,5,5,5,4,12,...
0,0,0,0,12,2.5,5,5,5,5,4,2.5,12,0,0,0,0,12,2.5,5,5,5,5,4,2.5];%通过弧需要的时间
edgecap=[0,0,0,0,0,0,19,33,33,33,33,25,77,0,0,0,0,77,19,33,33,33,33,25,77,...

0,0,0,0,77,19,33,33,33,33,25,19,77,0,0,0,0,77,19,33,33,33,33,25,19];%弧的容量

e=length(classroom)*num;%还未逃离的蚂蚁数
short=dshort(A,w);%每节点到最近的安全通道的距离
Eta=qifa(A,edgelen,short);%生成启发式矩阵
pheromoneMatrix=ones(50,1);%信息素矩阵初始化
edgeant=zeros(50,80);%每条弧上现有的蚂蚁
actant=[];%已经从教室出来但还未到达终点的蚂蚁编号
antfollow=zeros(e,3);%跟踪每只蚂蚁的情况
antfollow2=zeros(e,48);%记录每只蚂蚁的路径
[antfollow,antfollow2]=initant(antfollow,antfollow2,classroom,num);%antfollow初始化 起始节点为教室

%直到所有蚂蚁都逃出来
while(e~=0 && t<200)
    [t,antfollow,antfollow2,actant,pheromoneMatrix,edgeant,e]=newtime(t,A,classroom,num,antfollow,antfollow2,actant,w,p0,pheromoneMatrix,edgeant,Alpha,Beta,Eta,edgelen,e,bigroom,edgetime,edgecap);
end
```

```
function [t,antfollow,antfollow2,actant,pheromoneMatrix,edgeant,e]=newtime(t,A,classroom,num,antfollow,antfollow2,actant,w,p0,pheromoneMatrix,edgeant,Alpha,Beta,Eta,edgelen,e,bigroom,edgetime,edgecap)
%每0.5s进行一次更新
%t_时间 A_无向图的矩阵 classroom_教室节点 num_教室人数 antfollow_跟踪蚂蚁(所在弧的起始和末尾节点以及在弧上的剩余行进时间)
%antfollow2_储存每只蚂蚁的路径 actant_正在运行的蚂蚁 w_安全通道节点 pheromoneMatrix_信息素矩阵
%edgeant_储存在每条弧上运行的蚂蚁 Alpha_信息素影响因子 Beta_能见度加权值 Eta_启发式矩阵
%edgelen_每条弧得长度 e_未到达安全通道的蚂蚁 bigroom_大教室节点 edgetime_每条弧的旅行时间
T0=0.5;
T=t/T0+1;
judgepoint=zeros(48,8);%储存每个节点需要判断走向的蚂蚁
%actant 时间-0.5 判断它是否为0, 为0则加入judgepoint
for k=1:length(actant)
    n=actant(k);
    antfollow(n,3)=antfollow(n,3)-T0;
    if(antfollow(n,3)==0)
        us=arriveend(antfollow(n,2),w);
        if(us==1)%判断蚂蚁是否到达终点
            actant(k)=0;
            e=e-1;
        else
            judgepoint(antfollow(n,2),:)=nozeropush(judgepoint(antfollow(n,2),:),n);%没到终点则需要继续判断
            %从之前的弧出来
            s=findedge1(A,antfollow(n,1),antfollow(n,2));
            edgeant(s,edgeant(s)==n)=0;
            edgeant(s,:)=sort(edgeant(s,:),'descend');
            antfollow(n,1)=antfollow(n,2);
        end
    end
end
%删除已经到达终点的蚂蚁编号
actant(actant==0)=[];
%蚂蚁从教室出来
if(T<=num)
    for k=1:length(classroom)
```

```
34          antnum=(k-1)*num+T;%教室出来的蚂蚁编号1-40
35          pass=A(classroom(k),2);%教室对应的走廊节点
36          if(arriveend(classroom(k),bigroom))
37              antfollow(antnum,2)=pass;
38              antfollow2(antnum,2)=pass;
39              antfollow(antnum,3)=edgetime(A(classroom(k),3));
40              edge=A(classroom(k),3);
41              edgeant(edge,:)=nozeropush(edgeant(edge,:),antnum);
42          else
43              antfollow(antnum,1)=pass;
44              antfollow(antnum,2)=pass;
45              antfollow2(antnum,2)=pass;
46              judgepoint(pass,:)=nozeropush(judgepoint(pass,:),antnum)
    ;%需要判断走向.
47          end
48          actant=[actant antnum];%已经出来还没有走到终点的蚂蚁
49      end
50  end
51  %在节点的蚂蚁判断如何走
52  [antfollow,antfollow2,edgeant]=judge(judgepoint,A,classroom,
    pheromoneMatrix,Eta,Alpha,Beta,antfollow,antfollow2,edgecap,
    edgeant,w,edgetime);
53  %信息素更新
54  pheromoneMatrix=pheromone(pheromoneMatrix,edgeant,p0,antfollow2,
    edgelen,A);
55  t=t+T0;%时间+T0s
```

```
1  function [antfollow,antfollow2,edgeant]=judge(judgepoint,A,classroom
    ,pheromoneMatrix,Eta,Alpha,Beta,antfollow,antfollow2,edgecap,
    edgeant,w,edgetime)
2  %在节点的蚂蚁判断如何走
3  %judgepoint_储存每个节点带判断方向的蚂蚁编号 A_邻接矩阵 classroom_教
    室节点 pheromoneMatrix_信息素矩阵
4  %Eta_启发式函数 Alpha_信息素影响因子 Beta_能见度加权值antfollow_跟踪
    蚂蚁 antfollow2_储存每只蚂蚁的路径
5  %edgecap_储存每条弧的容量 edgeant_储存每条弧上的蚂蚁 w_安全通道
    edgetime_每条弧需要的通行时间
6  for k=1:length(judgepoint)%每个节点分别判断
7      d=judgepoint(k,:);%第k个节点需要judge的蚂蚁
8      d=sort(d,'descend');%让"楼上的蚂蚁先选择"
9      for i=1:nozero(d)
10         openlist=[];
```

```
11          openlist=open(k,antfollow2(d(i),:),A,classroom);%下一步能选
    择的点（不往教室和已走过的节点走）
12          x=[];
13          y=[];
14          y1=0;
15          y2=0;
16          %查看对应点之间的弧还能不能上去
17              for j=1:length(openlist)
18                  edge=findedge1(A,k,openlist(j));
19                  u=nozero(edgeant(edge,:))/edgecap(edge);
20                  y=[y u];
21                  %如果弧满了
22                  if(u>=1)
23                      x=[x j];
24                  end
25              end
26              if(length(x)==length(openlist))%如果弧全都满了
27                  %找最空的（比例小）挤一挤
28                  [y1,y2]=min(y);
29                  node=openlist(y2);
30              else%有弧没有满，则在没满的里面挑
31                  openlist(x)=[];
32                  node=pick_node(pheromoneMatrix,Eta,openlist,k,Alpha,
    Beta,w,A);
33              end
34              %选好了之后
35              antfollow(d(i),1)=antfollow(d(i),2);
36              antfollow(d(i),2)=node;
37              h=findedge1(A,k,node);%找到他们之间对应的弧
38              antfollow(d(i),3)=edgetime(h);
39              antfollow2(d(i),:)=nozeropush(antfollow2(d(i),:),node);
40              edgeant(h,:)=nozeropush(edgeant(h,:),d(i));
41      end
42  end
```

```
1  function pheromoneMatrix=pheromone(pheromoneMatrix,edgeant,p0,
       antfollow2,edgelen,A)
2  %信息素更新函数
3  %pheromoneMatrix_信息素矩阵 edgeant_储存每条弧上的蚂蚁 p0_信息素挥发
       因子
4  %antfollow2_储存每只蚂蚁的路径 edgelen_每条弧的长度 A_邻接矩阵
5  for k=1:length(pheromoneMatrix)
```

```
6       pheromoneMatrix(k)=pheromoneMatrix(k)*(1-p0);
7       for l=1:nozero(edgeant(k,:))
8           b=edgeant(k,l);
9           pheromoneMatrix(k)=pheromoneMatrix(k)+1/pathlen(antfollow2(b
        ,:),A,edgelen);
10      end
11  end
```

```
1   function [antfollow,antfollow2]=initant(antfollow,antfollow2,
        classroom,num)
2   %初始化
3   %antfollow_跟踪每只蚂蚁 antfollow2_储存每只蚂蚁的路径,classroom_教室
        节点 num_教室人数
4   for k=1:length(classroom)
5       for l=1:num
6           a=num*(k-1)+l;%蚂蚁编号
7           %初始节点为教室
8           antfollow(a,1)=classroom(k);
9           antfollow(a,2)=classroom(k);
10          antfollow2(a,1)=classroom(k);
11      end
12  end
```

```
1   function node=pick_node(pheromoneMatrix,Eta,openlist,s,Alpha,Beta,w,
        A)
2   %赌轮转盘选下一位置
3   %pheromoneMatrix_信息素矩阵 Eta_启发式函数 openlist_可选择的下一位置
        s_当前节点
4   %Alpha_信息素影响因子 Beta_能见度加权值 w_安全通道 A_邻接矩阵
5   i=length(openlist);
6   f=arriveend2(openlist,w);%判断openlist中是否有终点
7   if i==1
8       node=openlist(1);
9   elseif(f(1))%openlist中有终点，则直接选择终点。
10      node=f(2);
11  else
12  %赌轮转盘选择
13      PP=[];
14      for j=1:i
15          r=openlist(j);
16          u1=findedge1(A,s,r);
17          q=(pheromoneMatrix(u1)^Alpha)*(Eta(s,r)^Beta);
```

```
        PP=[PP q];
    end
    sumpp=sum(PP);
    PP(1)=PP(1)/sumpp;
    for j=2:i
        PP(j)=PP(j)/sumpp+PP(j-1);
    end
    q=unifrnd(0,1);%[0,1]的随机数
    for a=1:i
        if(PP(a)>=q)
            break;
        end
    end
    node=openlist(a);
end
```

```
function a=pathlen(path,A,edgelen)
%求某条路径的长度
%path_路径 A_邻接矩阵 edgelen_每条弧的长度
a=0;
for k=1:(nozero(path)-1)
    a=a+edgelen(findedge1(A,path(k),path(k+1)));
end

```

```
function openlist=open(node,closelist,A,classroom)
%可选择节点列表
%node_当前节点 closelist_已走过的节点 A_邻接矩阵 classroom_教室节点
openlist=[];
l=A(node,1);
for k=1:l
    openlist=[openlist A(node,2*k)];
end
%删除教室节点
b=intersect(openlist,classroom);
for k=1:length(b)
    openlist(openlist==b(k))=[];
end
%若只有一个点了，则允许往回走
if(~isequal(intersect(openlist,closelist),openlist))
    b=intersect(openlist,closelist);
    for k=1:length(b)
```

```
            openlist(openlist==b(k))=[];
        end
end

```

```
function s=findedge1(A,a,b)
%找到节点a,b对应的弧
%a,b_节点 A_邻接矩阵
m=[];
for u=1:A(a,1)
    m=[m A(a,2*u)];
end
h=find(m==b);
s=A(a,2*h+1);
```

```
function Eta=qifa(A,edgelen,short)
%生成启发矩阵
%A_邻接矩阵 edgelen_edgelen(i)为弧i的长度 short_short(i)为节点i到离
    其最近的门的最短距离
Eta=zeros(length(A));
for k=1:length(A)
    %m中储存与k节点相连的节点
    %n中储存对应弧
    m=[];
    n=[];
    for u=1:A(k,1)
        b=A(k,2*u);
        m=[m b];
        a=findedge1(A,k,b);
        n=[n a];
    end
for l=1:length(m)
    Eta(k,m(l))=1/(1+edgelen(n(l))+short(m(l)));
end
sumpp=sum(Eta(k,:));
Eta(k,m(1))=Eta(k,m(1))/sumpp;
for j=2:length(m)
    Eta(k,m(j))=Eta(k,m(j))/sumpp+Eta(k,m(j-1));
end
end
```

```
function a=nozeropush(a,n)
```

```
%在数组a第一个0处插入n
for k=1:length(a)
    if(a(k)==0)
        a(k)=n;
        break;
    end
end
```

```
function n=nozero(a)
%数组a中前n个非零数
n=0;
for k=1:length(a)
    if(a(k)==0)
        break;
    else
        n=n+1;
    end
end
```

```
function [mindistance,path] = mydijkstra(A,sb,db)

%  function [mindistance,mypath] = mydijkstra(A,sb,db);
%  输入: A邻接矩阵; A(i,j)i到j之间的距离。
%  sb起点, db终点。
%  mindistance最短路的距离, mypath最短路的路径

n = size(A,1);
visited(1:n) = 0;
distance(1:n) = inf;
distance(sb) = 0; %起点到各顶点距离的初始化
visited(sb) = 1;
u = sb; % u为最新的P标号顶点
parent(1:n) = 0; % 前驱顶点的初始化
for i = 1:n-1
    id = find(visited == 0); %查找未标号的顶点
    for v = id
        if A(u,v) + distance(u) < distance(v)
            distance(v) = distance(u) + A(u,v); %修改标号值
            parent(v) = u;
        end
    end
    temp = distance;
```

```
    temp(visited == 1) = inf; % 已标号点的距离换成无穷
    [t,u] = min(temp); %找标号值最小的顶点
    visited(u) = 1; %标记已经标号的顶点
end
path = [];
if parent(db) ~= 0 %如果存在路!
    t = db; path = db;
    while t ~=sb
            p = parent(t);
            path = [p path];
            t = p;
    end
end
mindistance = distance(db);
```

```
function short=dshort(A,w)
%求其距离最近安全通道的最短距离矩阵
%A_邻接矩阵 w_安全通道
%short_short(i)为节点i到离其最近的门的最短距离
short=zeros(length(A),1);
B=lingjie();
for k=1:length(A)
    min=1000;
    for l=1:length(w)
        [m,path]=mydijkstra(B,w(l),k);%到门的最短距离
        if(m<min)
            min=m;
        end
    end
    short(k)=min;
end
```

```
function a=arriveend(i,w)
%元素i是否在数组w中
a=0;
for k=1:length(w)
    if(w(k)==i)
        a=1;
        break;
    end
end
```

```
function f=arriveend2(openlist,w)
%判断openlist中是否有w中的元素
%输入: openlist 数组 w 数组
f=[0,0];
for k=1:length(openlist)
    if(arriveend(openlist(k),w))
        f(1)=1;
        f(2)=openlist(k);
        break;
    end
end
```

表 5: 蚁群算法对人员撤离路径的模拟

2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	9	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0
2	9	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0
2	9	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0
2	9	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0
2	9	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0
2	9	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0
2	9	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0
2	9	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0
2	9	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0

3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0	0

4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0
4	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0	0

5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0
5	12	11	10	9	8	0	0	0	0	0	0	0	0	0	0	0

7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
14	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0	0

14	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0	0
14	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0	0
14	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0	0
14	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0	0
14	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0
15	21	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0
15	21	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0
15	21	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0
15	21	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0
15	21	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0
15	21	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0
15	21	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0

15	21	20	19	30	42	43	44	45	46	47	35	24	13	0	0	0
15	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	30	42	43	44	45	46	47	35	24	13	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
16	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0

17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	30	42	43	44	45	46	47	35	24	13	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
17	23	22	21	20	19	8	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0

18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0

25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	32	33	34	35	47	46	45	44	43	42	30	19	8	0	0
25	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0
25	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0
25	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0
25	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0
25	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0
25	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0
25	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0
25	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
25	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0

26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0	0
26	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0
26	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0
26	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0
26	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0
26	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
26	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0
26	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0	0
26	32	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0

27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	42	43	44	45	46	47	35	24	13	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
27	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0

28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0
28	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0
28	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0
28	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0
28	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0
28	34	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
28	34	33	32	31	30	19	8	0	0	0	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0

29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	31	32	33	34	35	24	13	0	0	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
29	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0

36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0

37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
37	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0

38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
38	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0

39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
39	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0

40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
40	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0

41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	43	44	45	46	47	35	24	13	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
41	42	30	19	8	0	0	0	0	0	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0

48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	46	45	44	43	42	30	19	8	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0
48	47	35	24	13	0	0	0	0	0	0	0	0	0	0	0	0