《数据结构》课程设计总结

学号：1851573

姓名：李博宇

专业：计算机科学与技术

2020 年 7 月

目　录

第一部分 算法实现设计说明.....1
1.1 题目.....
1.2 软件功能.....
1.3 设计思想.....
1.4 逻辑结构与物理结构.....
1.5 开发平台.....
1.6 系统的运行结果分析说明.....
1.7 操作说明.....
第二部分 综合应用设计说明.....
2.1 题目.....
2.2 软件功能.....
2.3 设计思想.....
2.4 逻辑结构与物理结构.....
2.5 开发平台.....
2.6 系统的运行结果分析说明.....
2.7 操作说明.....
第三部分 实践总结.....
3.1.所做的工作.....
3.2.总结与收获.....
第四部分 参考文献.....

第一部分 算法实现设计说明

1.1 题目

给定一个有向图，完成：

(1)建立并显示出它的邻接链表；

(2)对该图进行拓扑排序，显示拓扑排序的结果，并随时显示入度域 的变化情况；

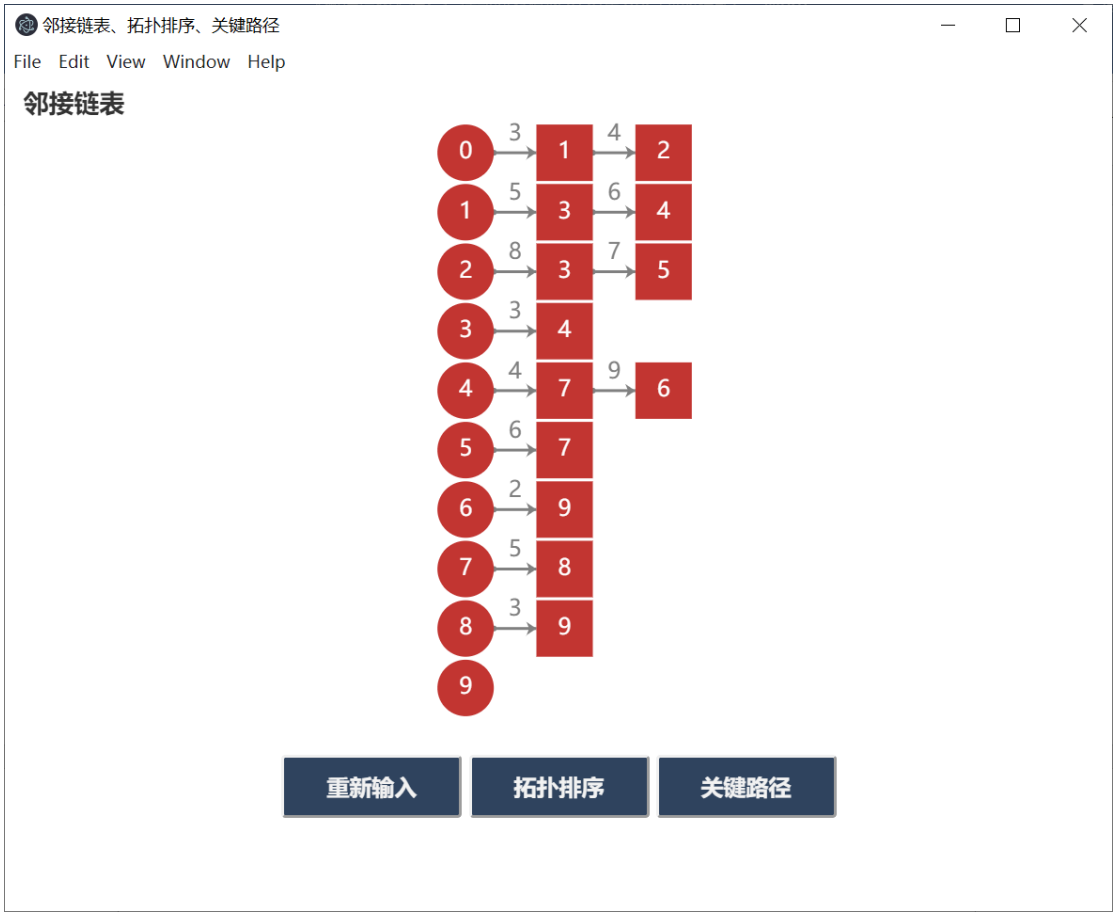
(3)给出它的关键路径（要求：显示出 Ve，Vl，E，L，L-E 的结果）。

1.2 软件功能

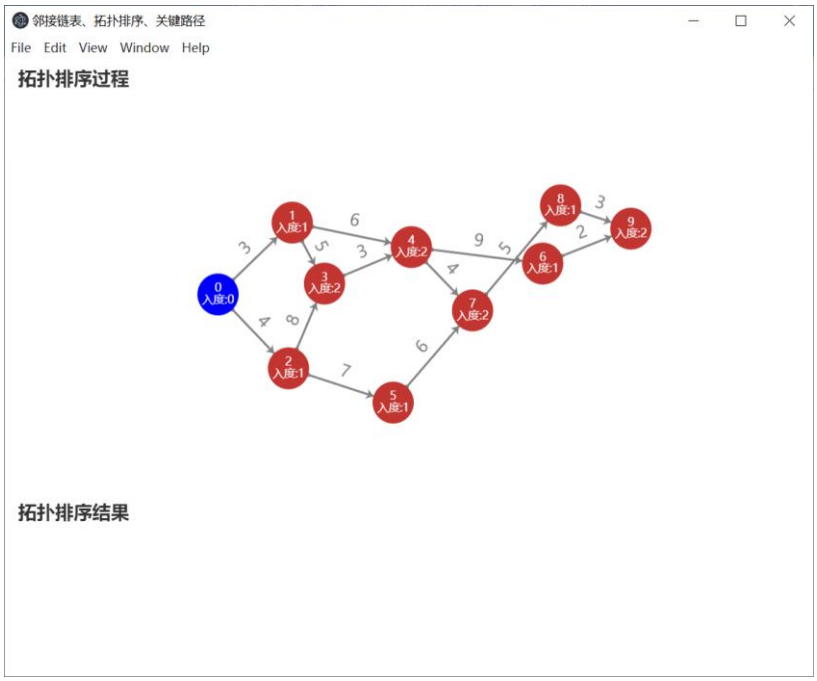
首先，软件的首要功能就是接受一个有向图的输入，之后才能进行邻接链表的显示以及拓扑排序与关键路径显示等功能。

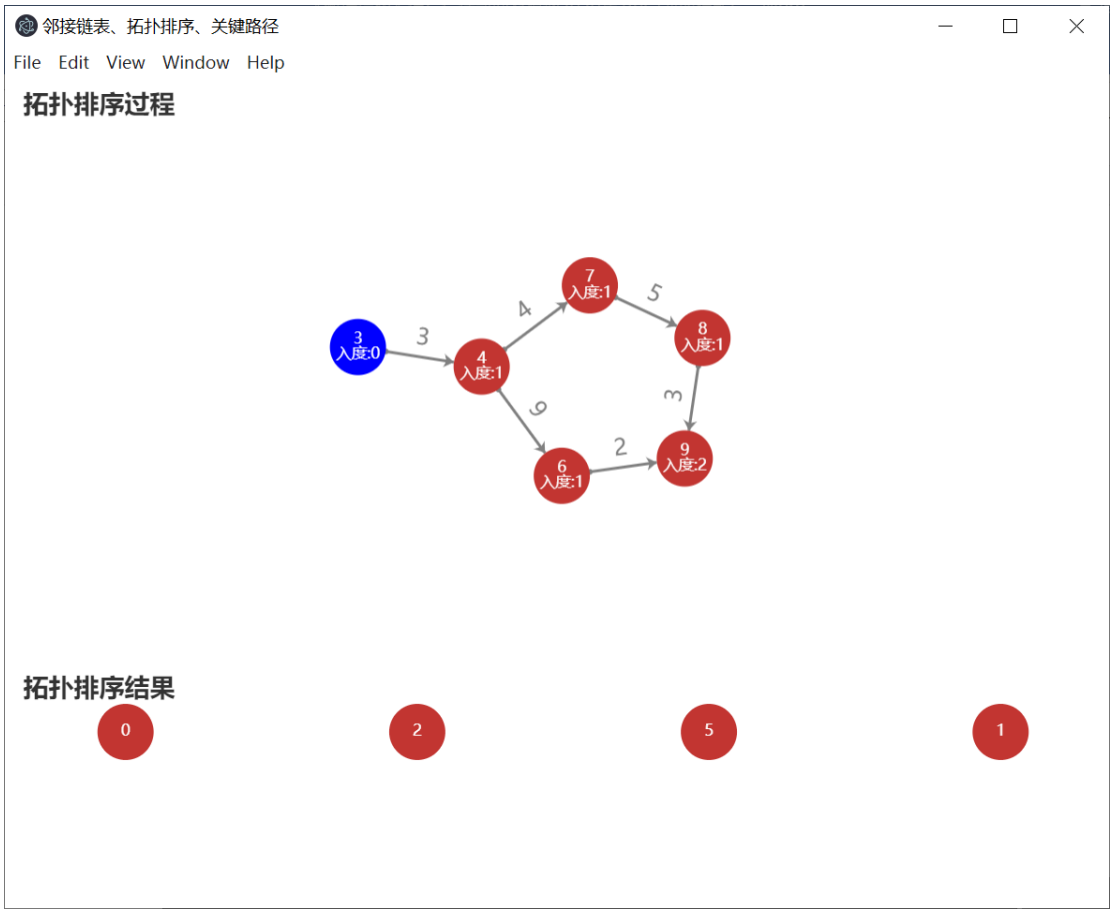
如图所示，该程序可以通过先输入节点，再输入边的形式输入一个有向图。

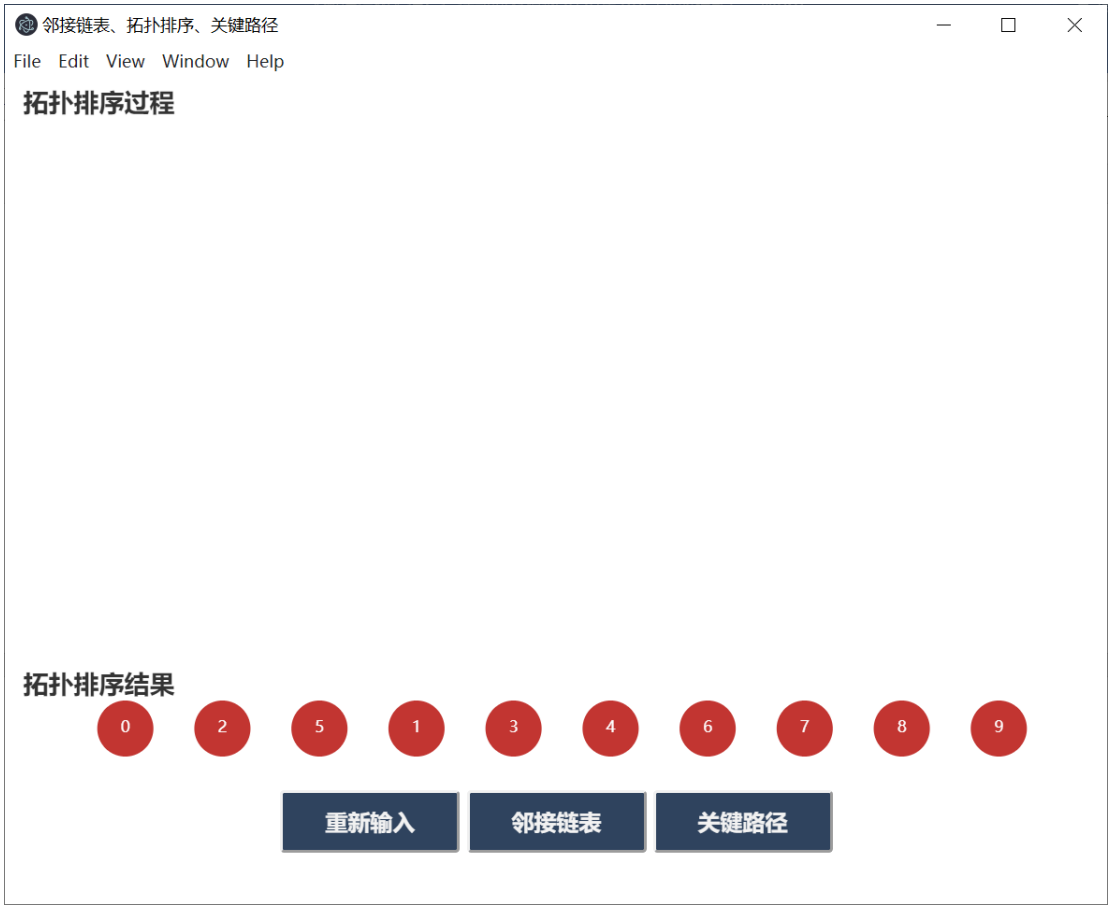
之后，可以将邻接链表以动画的形式显示出来，大概的实现方法是对邻接链表进行遍历，并且将相应的节点与边的信息储存到图形显示配置项[1]当中，进行显示。



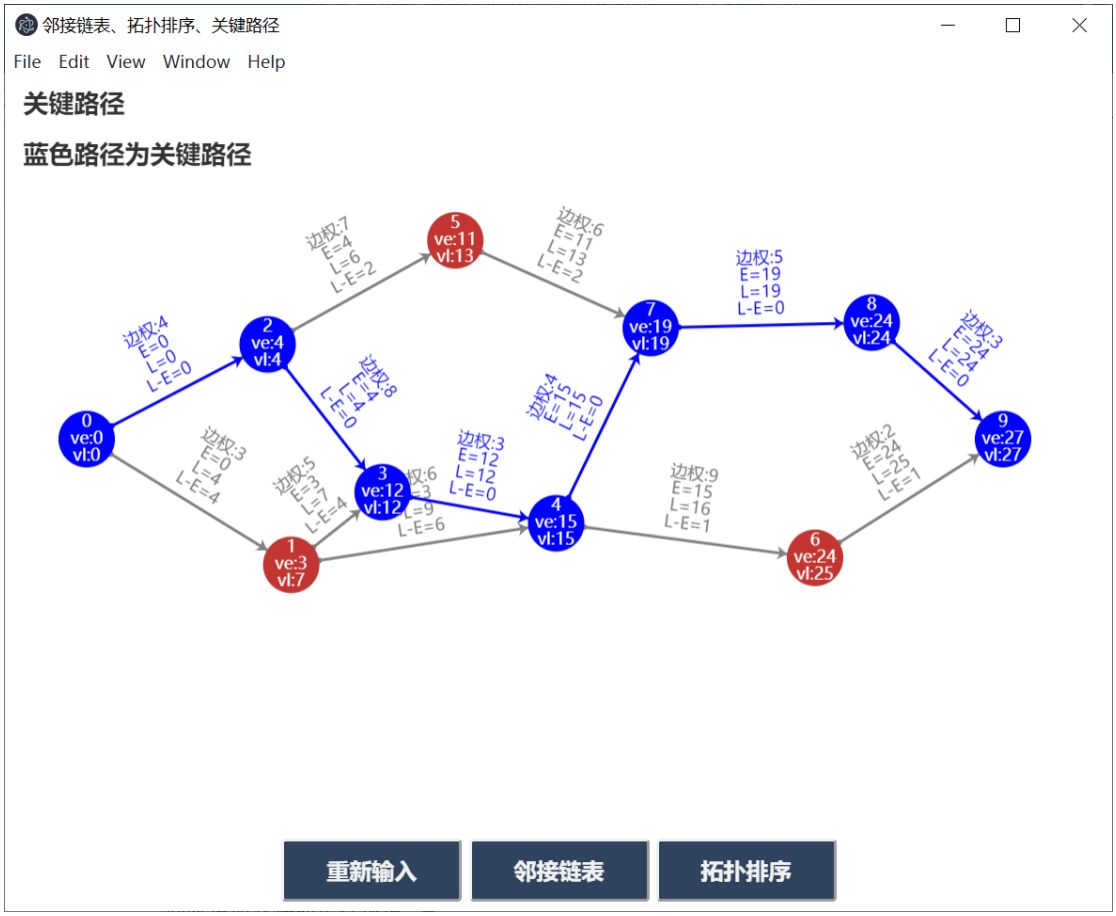
其次，该程序可以根据给出的邻接链表进行拓扑排序，并且能够以动画的形式将拓扑排序的过程显示出来。具体的实现方法就是在进行拓扑排序的同时将每一轮的排序结果都加载到动画显示配置项中，之后进行显示，而每一轮排序中间则是设置了延迟，确保能够出现动画效果，动画的延迟可以由外部输入决定。




邻接链表、拓扑排序、关键路径
File Edit View Window Help
拓扑排序过程
3
入度:0
4
入度:1
7
入度:1
8
入度:1
6
入度:1
9
入度:2
3
4
5
9
3
2
拓扑排序结果
0
2
5
1


邻接链表、拓扑排序、关键路径
File Edit View Window Help
拓扑排序过程
拓扑排序结果
0
2
5
1
3
4
6
7
8
9
重新输入
邻接链表
关键路径

最后，该程序还能够显示输入图的关键路径，并且显示各顶点的 Ve，Vl 以及各个边（活动）的 L 和 E 的值。具体的实现方法也是对输入得到的邻接矩阵实现关键路径算法，计算出各个顶点的最早开始时间以及最晚开始时间，进而计算出各个边上的最早和最晚开始时间，并且计算出关键路径，并且将相关的信息存储到图像显示的配置项当中，从而完成对关键路径的显示。



1.3 设计思想

整个算法实现题的其实按照题目的顺序来就是一个比较完整的实现思路。

首先是读取邻接链表，根据给定的节点信息确定节点的数量以及名称，开辟存储空间。之后读取边的信息，对于给定的每一条边，找到它的源点，在其后的链式结构中添加相应的汇点以及边权，完成邻接链表的构建。

```
//进行邻接矩阵的显示
function show_adjacency_list() {…
}
```

拓扑排序的过程，首先是创建存储各个节点入度的数组，并将其入度都初始化为 0。之

后遍历整个邻接表，对于每条边上出现的汇点，都相应的将其入度+1，得到每个节点的入度情况。遍历入度表，找到所有入度为 0 的节点，将节点名称加入栈中。

由于拓扑排序每次都会将一个节点从原图中剔除，所以对其进行完整的拓扑排序的次数一定是固定的，等于图中节点的个数。在第一次进行拓扑排序时，栈中应该是有初始入度为 0 的节点，之后弹出栈顶元素，然该元素所指的其他的节点的入度减一，并且监测入度改变，如果该节点入度变为了零，则将该节点也加入栈中，这样做保证了寻找其他的入度为 0 的点不会产生额外开销。之后每一次循环都重复此操作，即可完成整个拓扑排序的过程。其中，如果在进行某一次拓扑排序时，发现栈已空，在上一轮排序中没有产生出入度为 0 的节点，则说明该图无法产生有效的拓扑排序，则跳出循环，弹出错误显示框。[2]

```
//进行完整的拓扑排序
function show_topological_sort() {…
}

//根据节点的度信息进行一轮拓扑排序
function do_sort(degree_node, adjacency_list, node_stack, show) {…
}
```

在完成拓扑排序之后，需要实现的是关键路径算法。关键路径简单来说就是进行了一次正拓扑排序以及一次逆拓扑排序，在进行正拓扑排序的同时计算出各个顶点的最早发生时间，再通过逆拓扑排序计算出各个顶点的最晚发生时间。之后再遍历整张图，根据节点信息计算出每个边（活动的）最早发生时间以及最晚发生时间，并据此确定出该活动是否为关键活动，并且将计算出的信息都存入图形显示配置项中，进行显示。[3]

```
function show_critical_path() {…
}
```

1.4 逻辑结构与物理结构

逻辑结构：

在进行图存储时，采用的逻辑结构为图形结构，图为有向图，每个节点都可能有多个前驱节点和多个后继节点，是多对多的图形结构。

在进行拓扑排序中，利用了栈来存储所有的入度为 0 的节点，栈的开始节点与终端节点都是唯一的，每个节点有且只有一个前驱节点，有且仅有一个后继节点，属于线性结构。

在拓扑排序中存储各个节点的度关系，以及关键路径中存储各点以及各边的最早最晚发生时间时，则是使用了集合结构，各个节点之间并没有明确的逻辑关系，只是为了反映各个节点与其对应值的关系。

物理结构：

对于邻接矩阵的存储来说，首先是利用了散列存储结构，形成了节点名称到所指向节点链的映射。之后某一结点所指向的所有节点则是利用了链式存储结构，方便了频繁的插入删除。

对于存储入度为 0 节点的栈来说，由于主要是会频繁产生插入和删除操作，并且我们只关心尾部的节点情况，因此利用链式存储结构实现。

对于存储节点与其度的对应关系以及节点与边和最早最晚发生时间的数据结构来说，由于我们只关心对应值的情况，因此是使用散列存储结构进行存储。[4]

1.5 开发平台

开发平台：

Electron 8.0.3 GitHub 发布的跨平台桌面应用开发工具，支持 Web 技术开发桌面应用，其本身是基于 C++ 开发的，GUI 核心来自于 Chrome，而 JavaScript 引擎使用 v8。[5]

第三方库：

ECharts 一个使用 JavaScript 实现的开源可视化库，可以流畅的运行在 PC 和移动设备上，兼容当前绝大部分浏览器（IE8/9/10/11，Chrome，Firefox，Safari 等），底层依赖轻量级的矢量图形库 ZRender，提供直观，交互丰富，可高度个性化定制的数据可视化图表。[6]

jQuery 一个快速、简洁的 JavaScript 框架。 jQuery 设封装 JavaScript 常用的功能代码，提供一种简便的 JavaScript 设计模式，优化 HTML 文档操作、事件处理、动画设计和 Ajax 交互。[7]

软件的运行环境：Windows10

1.6 系统的运行结果分析说明

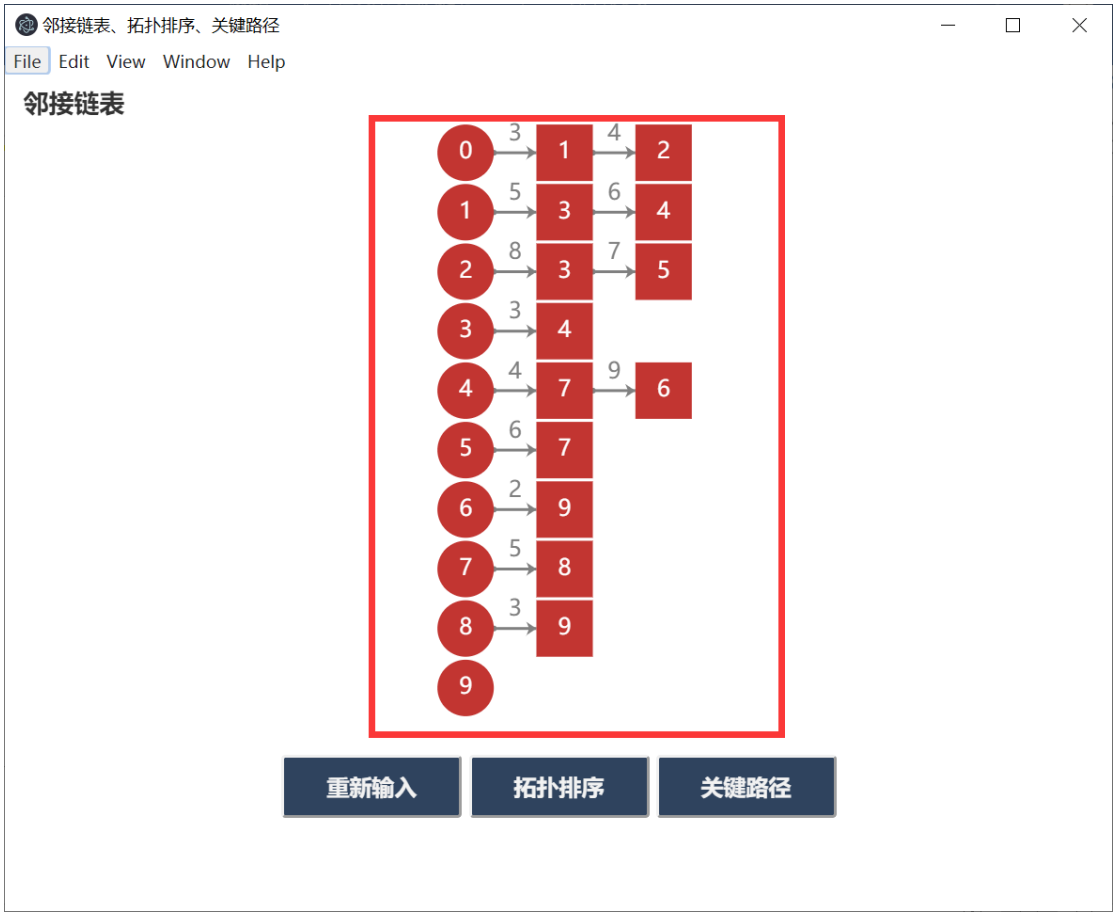
调试与开发过程：首先是对题目进行了仔细地阅读与思考，对于整个的算法流程以及数据结构设计有了一个大致的构想。之后便是搜索相关库与相关资料，进行开发。由于项目采

用的是基于 Electron 进行开发，因此整个项目的开发过程实际上就是在进行前端开发。

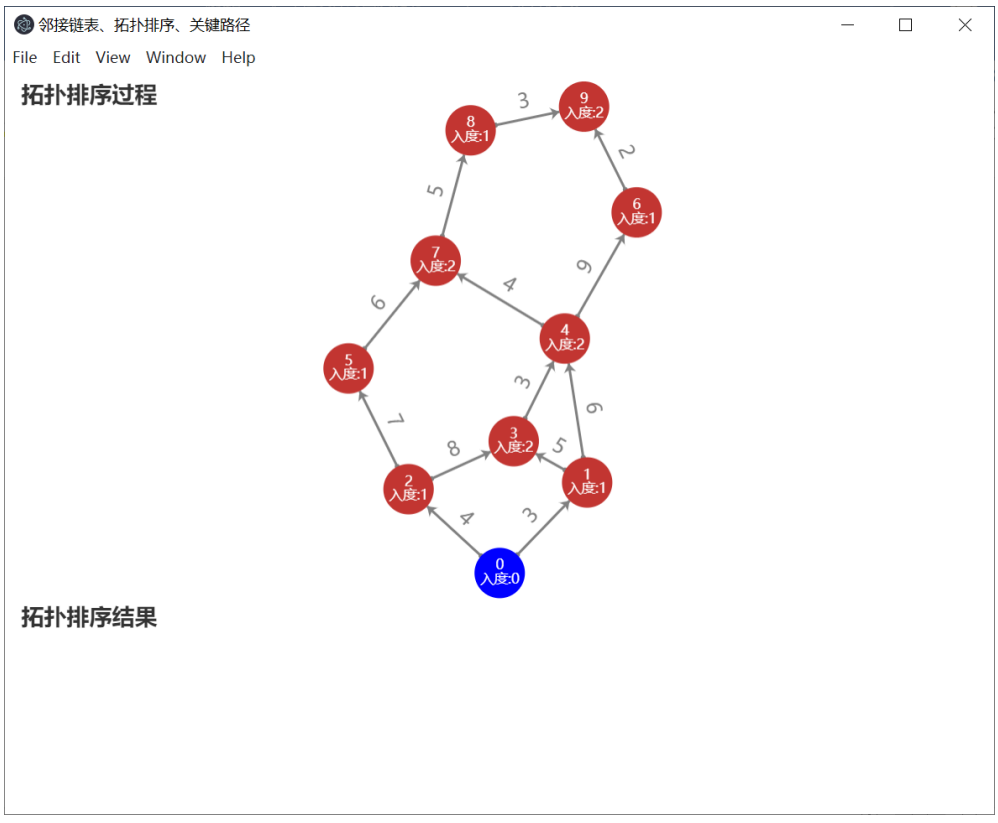
调试工具则是利用 Electron 中集成的类似浏览器的调试控制台进行调试。

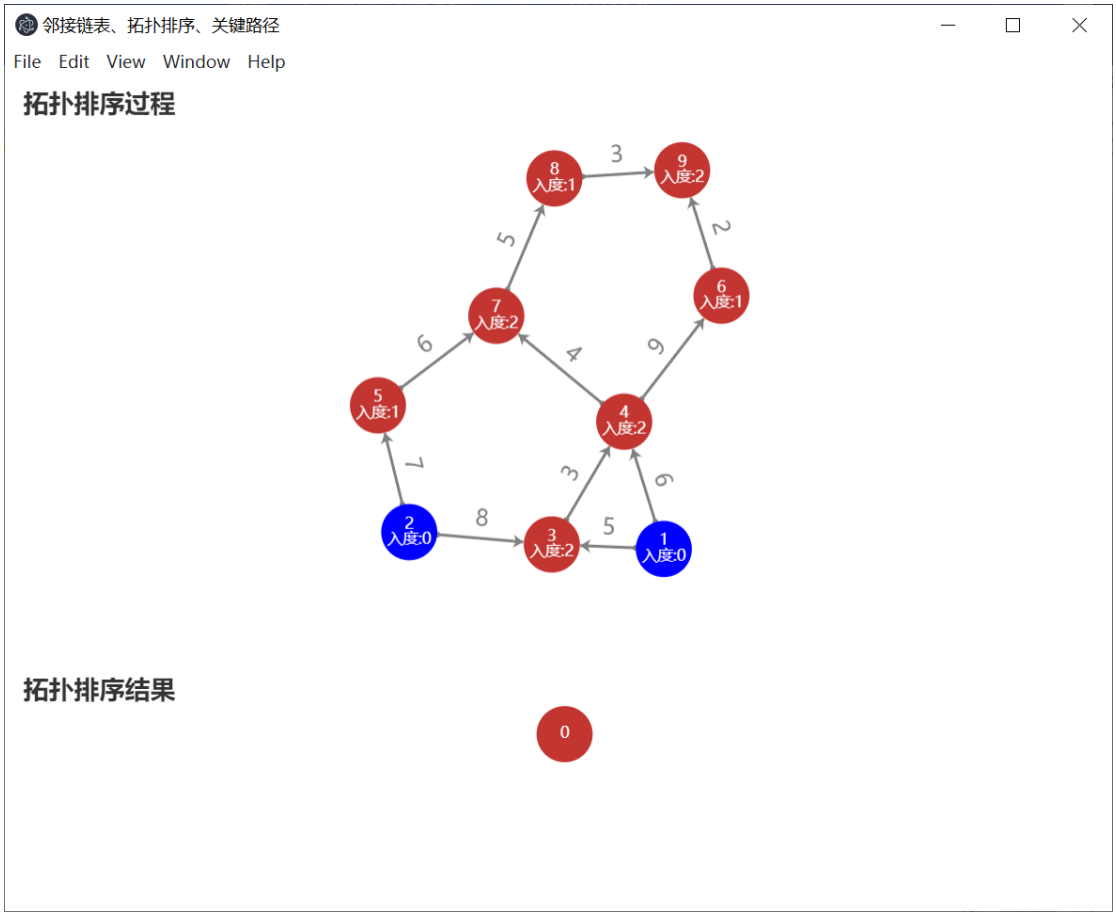
运行结果说明：首先可以打开程序可以看到输入界面，可以进行节点以及边的情况的输入。在程序打开之后是有默认的预置的值，当然也可以进行自行重新输入以及调整。

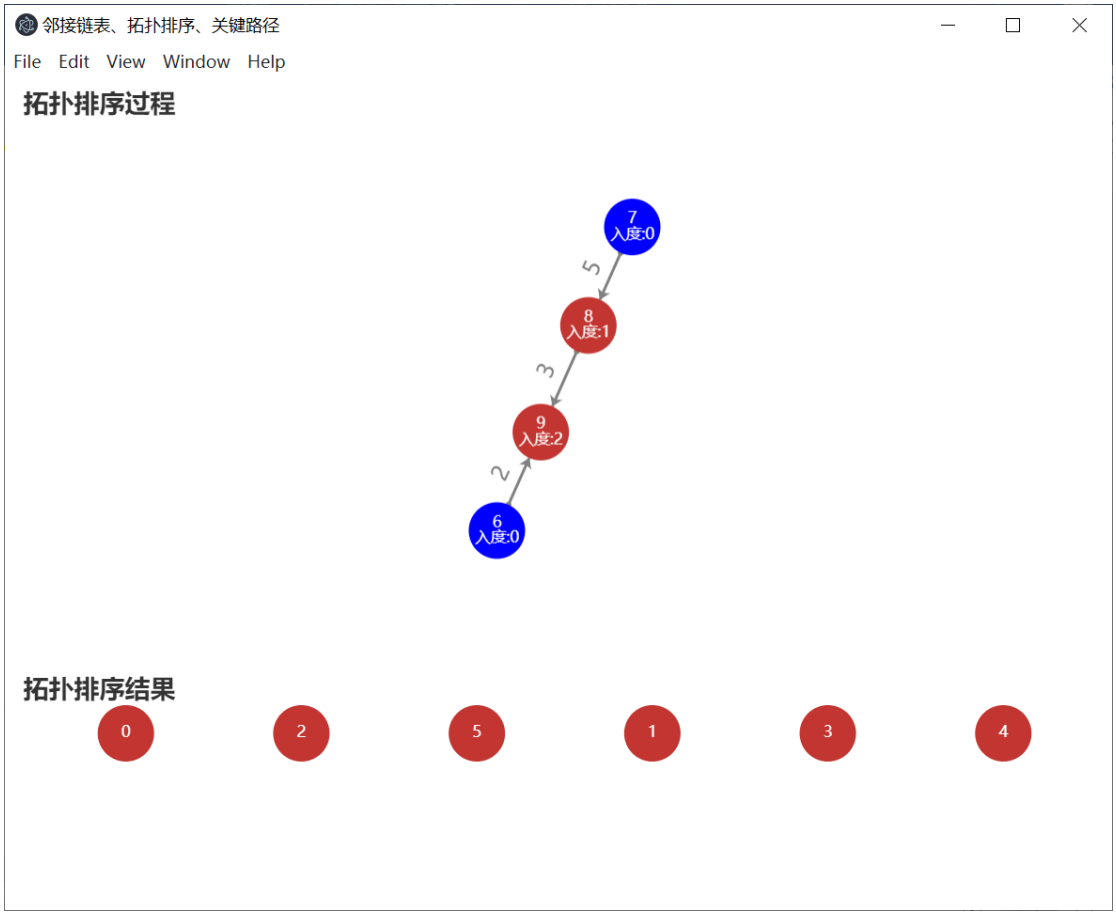
点击“邻接链表”按钮，可以显示输入图形的邻接链表：

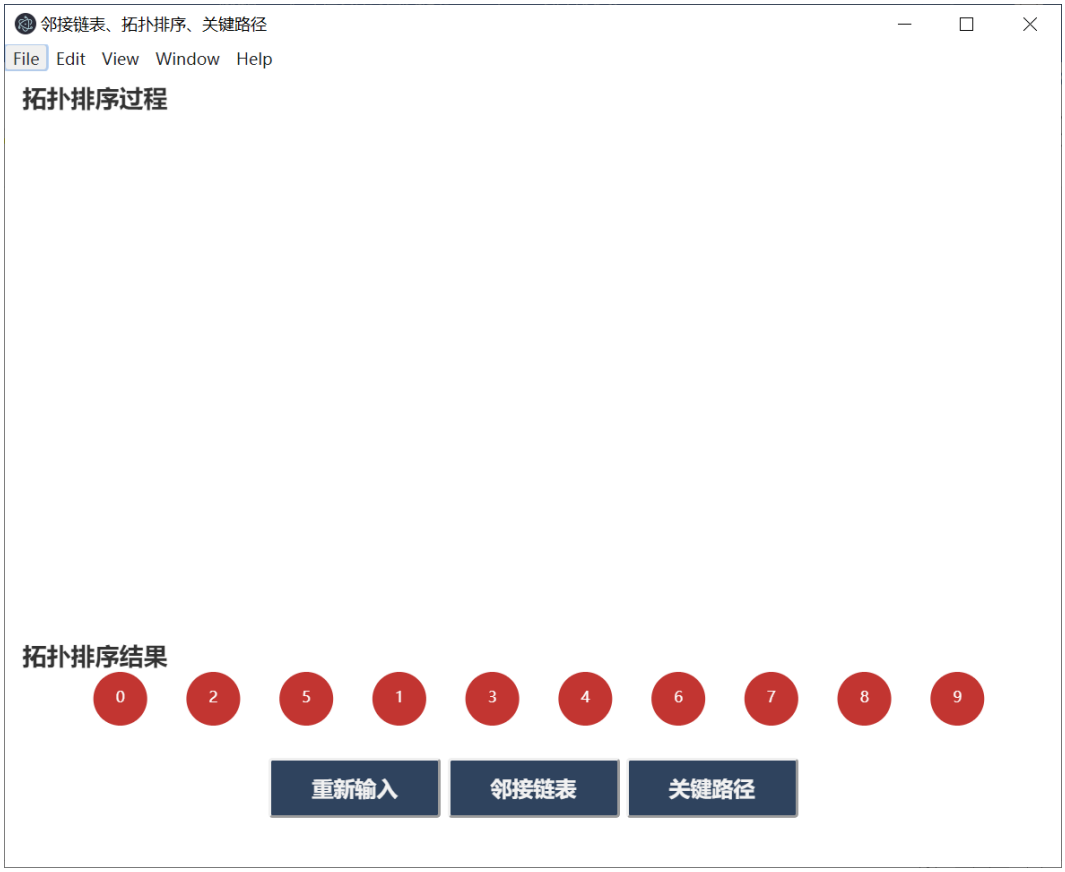


点击“拓扑排序”按钮，可以显示对给定图的拓扑排序的动画

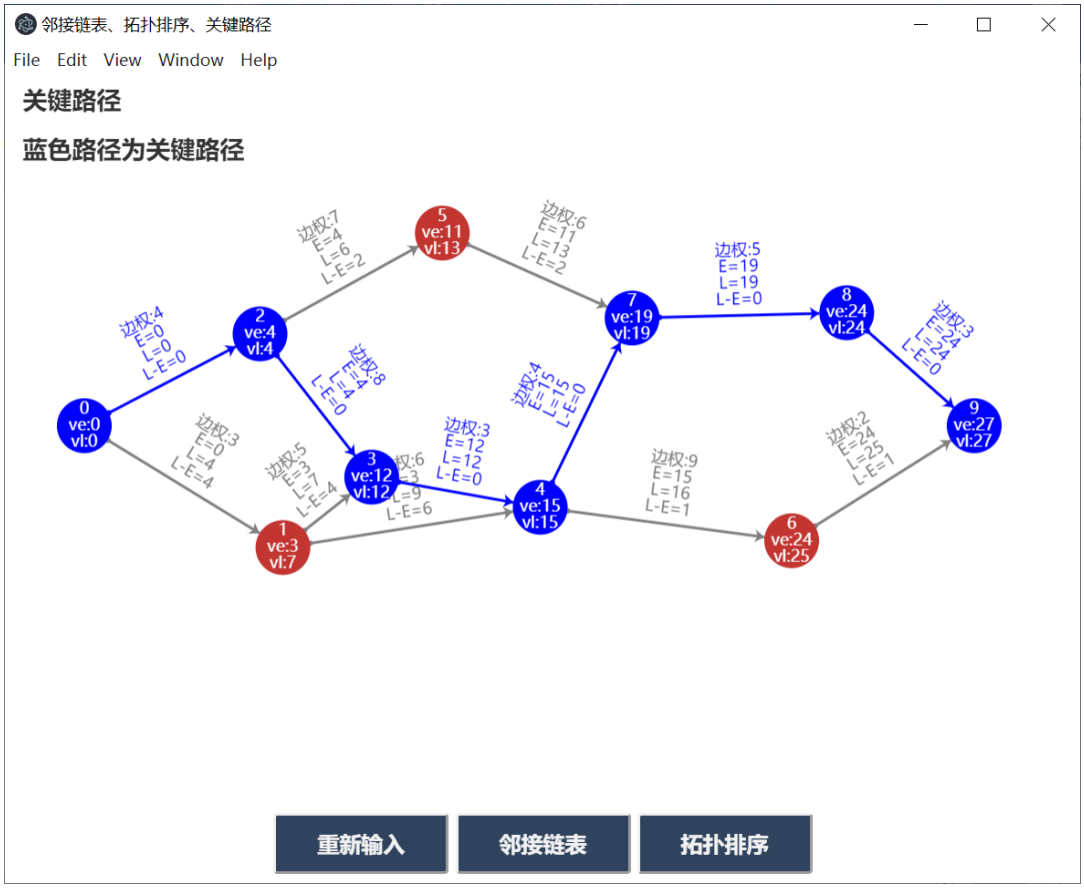



邻接链表、拓扑排序、关键路径
File Edit View Window Help
拓扑排序过程
8 入度:1
9 入度:2
7 入度:2
6 入度:1
5 入度:1
4 入度:2
2 入度:0
3 入度:2
1 入度:0
3
2
5
6
4
9
7
8
5
3
9
拓扑排序结果
0


邻接链表、拓扑排序、关键路径
File Edit View Window Help
拓扑排序过程
7 入度:0
8 入度:1
9 入度:2
6 入度:0
5
3
2
拓扑排序结果
0
2
5
1
3
4



点击“关键路径”按钮之后，则会显示当前图的关键路径：



当进行错误输入时，进行邻接链表以及拓扑排序关键路径等操作时将会显示错误提示

当输入的图的情况不满足拓扑排序条件时，即构成了回环时，则会报错并且提示无法进行拓扑排序或者关键路径生成操作。

1.7 操作说明

在打开程序以后，会自动进入输入界面，可输入得内容主要有图中的节点个数、各个节点的编号、边的个数、各个边的情况以及拓扑排序动画的显示延迟。

各输入框的输入规则如下：

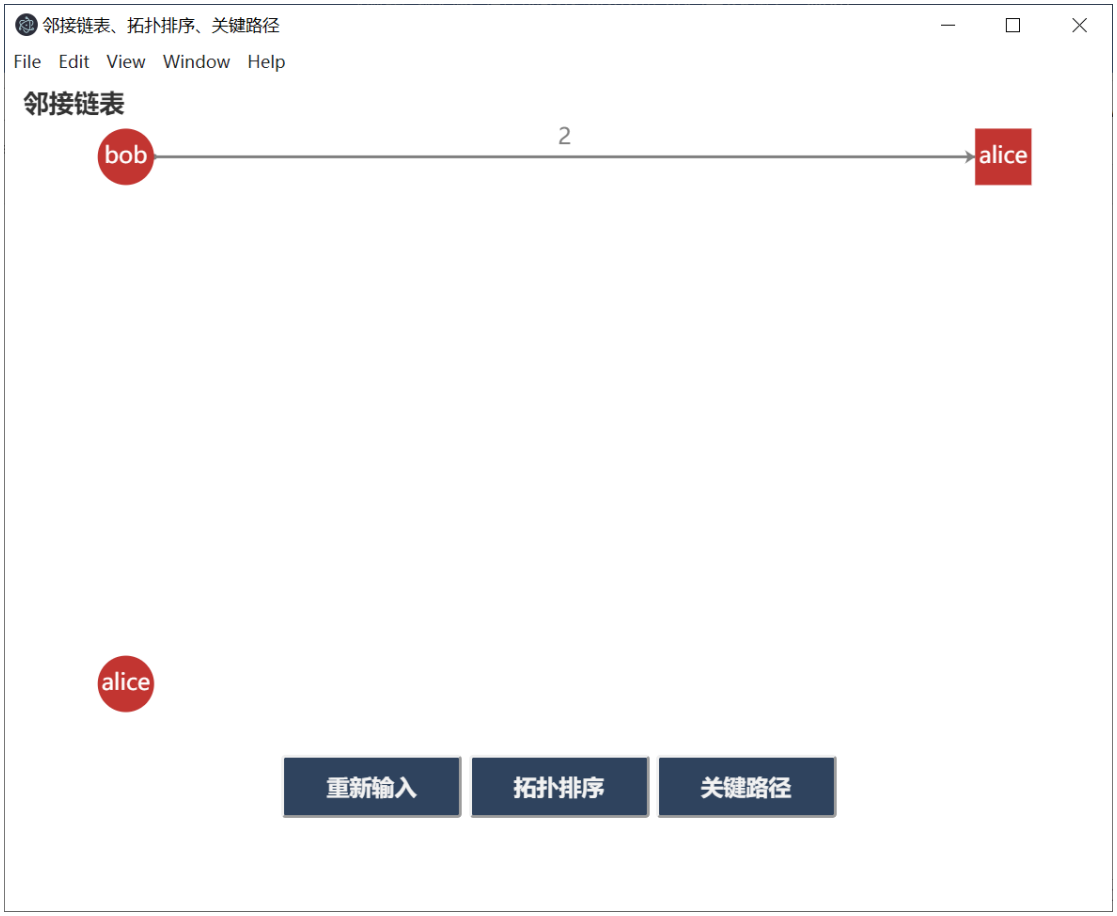
1. 输入节点个数要求是一个阿拉伯数字。

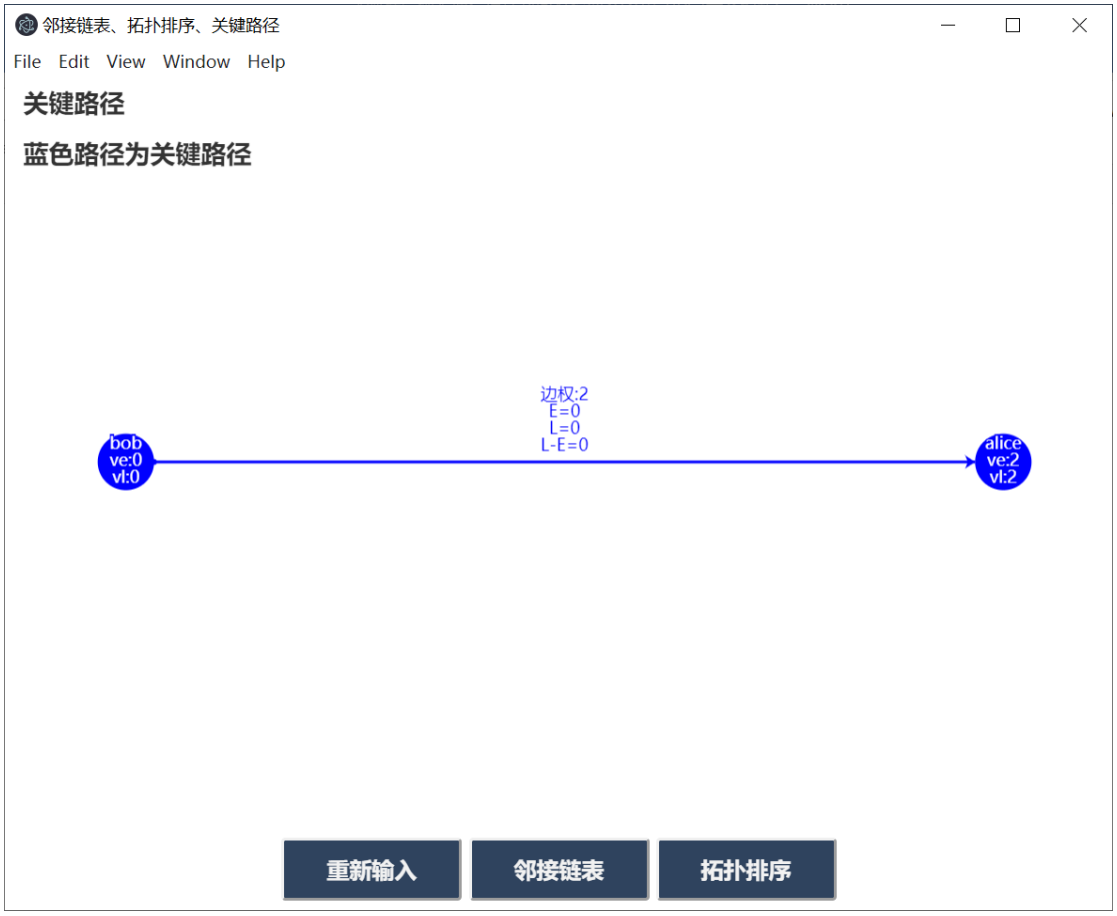
2. 输入的节点数量应当余输入的节点个数保持一致，输入节点过少可能引发错误，输入节点过多后续节点将不会被读取。各个节点编号之间用空格隔开，节点编号支持任意字符串，不仅限于数字。

3. 输入的边的个数要求是一个阿拉伯数字。

4. 输入边的信息时，每行输入一条边的信息，每条边的形式如下：u v w，u 表示源点的节点名称，v 表示汇点的节点名称，w 表示边的权重，三个数据之间需要用空格隔开。u v 需要都是在输入的节点名称中出现过的名称，边的数量需要和上面输入的数量相同。

输入完成后，即可点击相应按钮显示相应动画。


邻接链表、拓扑排序、关键路径
File Edit View Window Help
邻接链表
bob
2
alice
alice
重新输入
拓扑排序
关键路径


邻接链表、拓扑排序、关键路径
File Edit View Window Help
关键路径
蓝色路径为关键路径
边权:2
E=0
L=0
L-E=0
bob
ve:0
vl:0
alice
ve:2
vl:2
重新输入
邻接链表
拓扑排序