Graph实验报告

题目：矩阵相加和相乘

班级：2020211306 姓名：马天成 学号：2020211376 分工：撰写代码+实验报告

班级：2020211306 姓名：黄洪建 学号：2020211371 分工：撰写代码

班级：2020211306 姓名：马紫薇 学号：2020211392 分工：撰写实验报告

# 一：需求分析

## 题目的目的

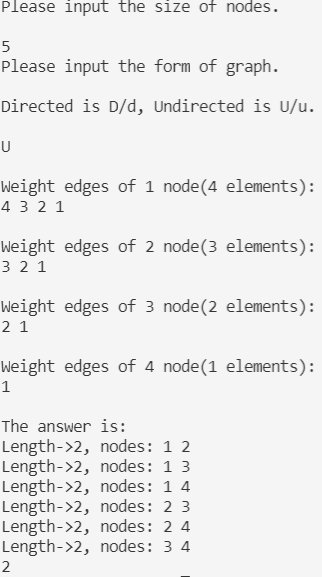
输出图的最短距离中的最长。

## 输入的正确形式

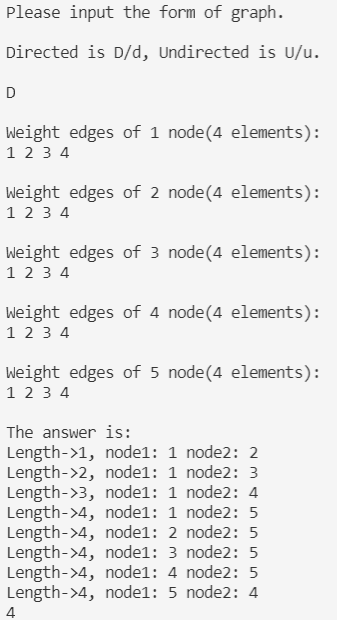
首先，永远不能相信用户的输入。

并且，我们对于溢出的数据会报错并要求重新输入。

无向图：



有向图：



Input:

1. size of nodes ：确定顶点个数；

2. U/u 或者 D/d ：确定图的有向或无向；

​ Undirected graph: 输入size-1行数字，以空格分开；每行输入（size-1-当前行数）个数字 ；

​ Undirected graph: 输入size行数字，以空格分开；每行输入（size-1-当前行数）个数字 ；

​ 其中，数字应该为有效；规定0为无穷，不可出现负数。

​ 有向图需要每行的size-1个边；

​ 无向图只需要按照顺序录入上半矩阵的边。

​

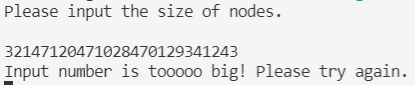
Output:

1. 最短路径遍历结果；

2. 数字答案。

## 输入的错误形式（我们对健壮性极其重视）

**1.溢出**

****



**2. 负数**



3. 个数少了



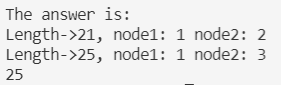
4. 个数多了



## 输出样例

**1. 正确输入形式**

**有向图输出：**



无向图输出：

# 二: 概要设计

## 问题解决思路

1.从键盘输入数据

2.同时建立邻接表和邻接矩阵

3.计算最短边Floyed

4.遍历输出

## 主函数伪代码

int main()

{

请输入顶点个数；

选择有向无向；

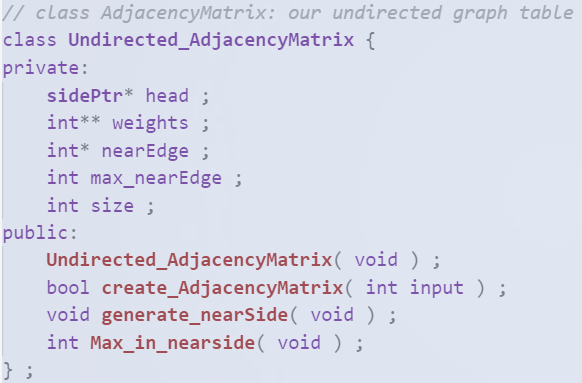
class matrix ; // 建立树（先建表，后建矩阵）

输出；

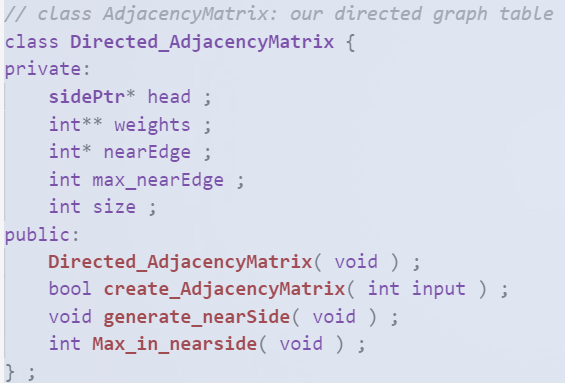
}

## class matrix

无向图



有向图

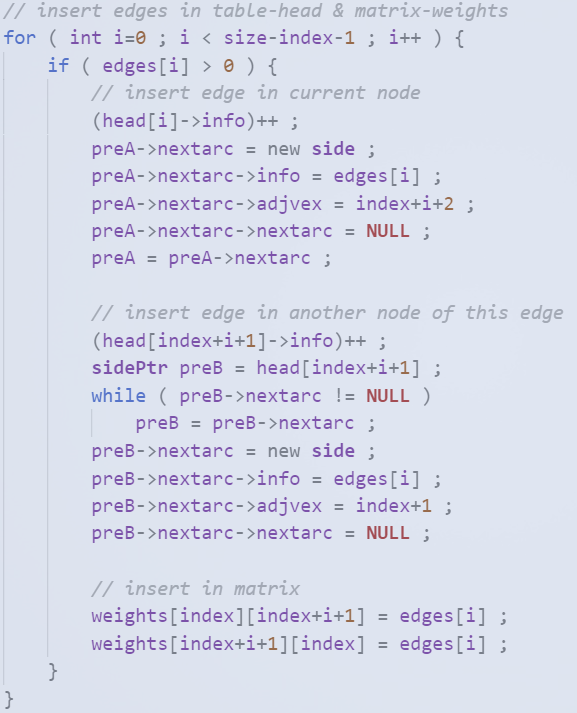


**建立过程**

**有向图：无难度，每行扫进去；**

**无向图：在邻接表建立过程中，发现我们需要相对称的建立链表节点。**

**在代码中你会发现我神奇的写法：**

****

重要的地方：

头节点计数；

邻接表插入时要进行对称插入，要找到行尾链表进行插入。

代码优化，判断有效且极少。

## 函数调用关系

int main()

obj.**Max\_in\_nearside**()

obj.**create\_AdjacencyMatrix**( size )

**返回答案**

obj.**generate\_nearSide**() ;

遍历输出相应节点

道歉：由于时间太少，把时间花在了代码上，还麻烦耐心看我写的代码；

我的代码应该不算难看懂，我写的应该挺规范的。