**南京信息工程大学算法设计与分析（实习）报告**

实验名称 日期 2023.6 得分 指导教师 赵振杰 专业 计科

学院 计网院 年级 2021 班次 5 姓名 刘祥宇 学号 202183290006

1. **实验目的**

1．实验目的

（1）熟悉图算法中的多种单源最短路径求解方法

（2）理解基于图的多种单元最短路算法的设计原理

（3）编写能够求解单源最短路径问题的图算法程序

（4）能够分析解题所用算法的时间复杂度

1. **实验内容**

有一个无向图，共 N 个节点，编号 1 至 N，共 M 条边。FJ 在节点 1，它想到达节点 N。FJ 总是会选择最短路径到达节点 N。作为捣蛋的奶牛 Bessie，它想尽量延迟 FJ 到达节点 N 的时间，于是 Bessie 决定从 M 条边之中选择某一条边，使得改边的长度变成原来的两倍，由于智商的问题，Bessie 不知道选择加倍哪条边的长度才能使得 FJ 到达 N 号节点的时间最迟。注意：不管 Bessie 选择加倍哪条边的长度，FJ 总是会从 1 号节点开始走最短路径到达 N 号点。

输入描述

第一行，两个整数 N 和 M. 1 <=N<=250, 1<=M<=250000。

接下来有 M 行，每行三个整数：A，B，L，表示节点 A 和节点 B 之间有一条长度为 L 的无向边。1<=L<=1000000。

输出描述

一个整数。Bessie 选择了加倍某一条边的长度后，奶牛 FJ 从节点 1 到达节点 N 的最短路径是多少。但是输出的格式有变化，假设 Bessie 没有加倍某一条边的长度之前，FJ 从 1 号节点到达 N 号节点的最短路径是 X；在 Bessie 加倍某一条边的长度之后，FJ 从 1 号节点到达 N 号节点的最短路径是 Y，那么你输出的结果是 Y-X。

样例输入

5 7

2 1 5

1 3 1

3 2 8

3 5 7

3 4 3

2 4 7

4 5 2

样例输出

2

1. **实验步骤和结果**

**#include<iostream>**

**#include<cstdio>**

**#include<algorithm>**

**#include<cstring>**

**#include<cmath>**

**#include<map>**

**#include<queue>**

**#include<set>**

**#define ll long long**

**#define llu unsigned ll**

**using namespace std;**

**const int mod=1000000007;**

**const int maxn=50100;**

**const int maxx=300;**

**const int inf=0x3f3f3f3f;**

**int head[maxx],edge[maxn],ver[maxn],nt[maxn];**

**int d[maxx];**

**int ha[maxx];**

**int pre[maxx];**

**int tot=1,n,m;**

**bool flag=false;**

**void add(int x,int y,int z)**

**{**

**ver[++tot]=y,edge[tot]=z;**

**nt[tot]=head[x],head[x]=tot;**

**}**

**int Dij(void)**

**{**

**memset(d,0x3f,sizeof(d));**

**memset(ha,0,sizeof(ha));**

**d[1]=0;**

**priority\_queue<pair<int,int> >q;//默认为大根堆**

**q.push(make\_pair(0,1));**

**while(q.size())**

**{**

**int x=q.top().second;**

**q.pop();**

**if(ha[x]) continue;**

**ha[x]=true;**

**for(int i=head[x];i;i=nt[i])**

**{**

**int y=ver[i],z=edge[i];**

**if(d[y]>d[x]+z)**

**{**

**d[y]=d[x]+z;**

**if(!flag) //最开始算最短路时，记上y点在最短路上的父亲边是哪条边**

**pre[y]=i;**

**q.push(make\_pair(-d[y],y));**

**}**

**}**

**}**

**return d[n];**

**}**

**int main(void)**

**{**

**scanf("%d%d",&n,&m);**

**int x,y,z;**

**for(int i=1;i<=m;i++)**

**{**

**scanf("%d%d%d",&x,&y,&z);**

**add(x,y,z);**

**add(y,x,z);**

**}**

**int cnt=Dij();**

**flag=true;**

**int \_max=0;**

**for(int i=2;i<=n;i++)//枚举点**

**{**

**int k=pre[i];**

**if (k==0)//如果不在最短路径上就不管了**

**continue;**

**edge[k]=edge[k]\*2,edge[k^1]=edge[k^1]\*2;**

**\_max=max(\_max,Dij());**

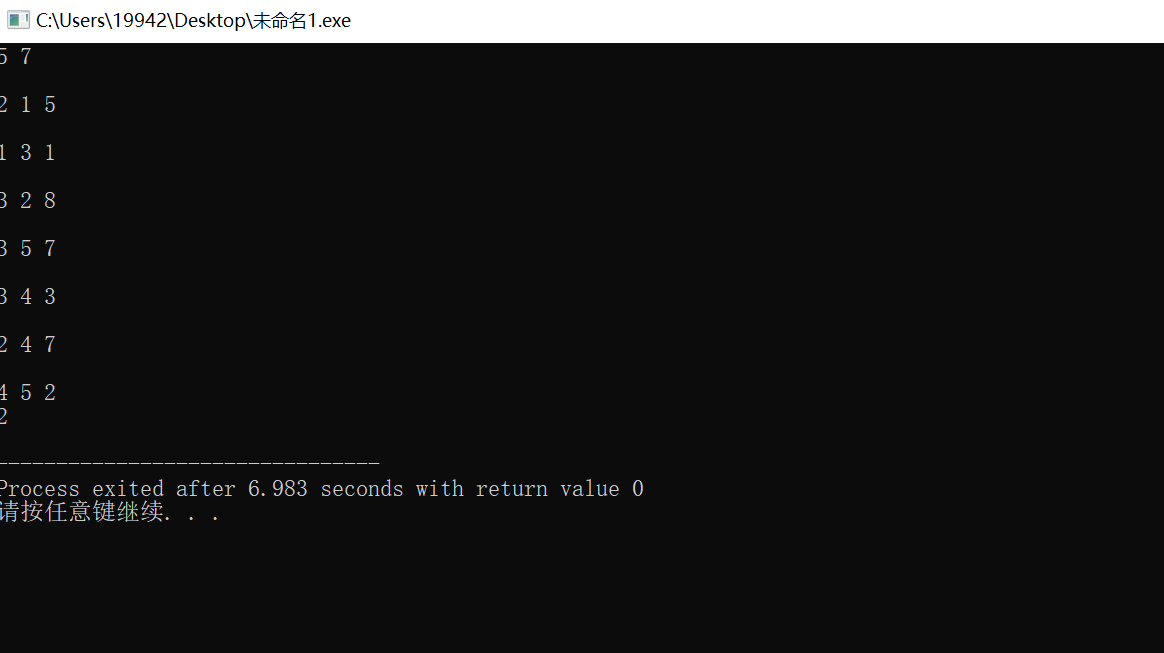
**edge[k]=edge[k]/2,edge[k^1]=edge[k^1]/2;**

**}**

**printf("%d\n",\_max-cnt);**

**return 0;**

**}**

****

1. **总结**

算法时间复杂度为O(E+VlgV**)**