**Ve 281 P5**

**Dong Jing**

**515370910182**

Here is the source code of main.cpp

#include <iostream>

#include <vector>

#include <queue>

#include <map>

**using** **namespace** std**;**

struct edge

**{**

int start**;**

int end**;**

int weight**;**

**};**

struct node

**{**

int dist**;**

node**\*** prev**;**

int in**;**

bool used**;**

int parent**;**

int rank**;**

int self**;**

int d**;**

bool reached**;**

**};**

struct compare

**{**

bool **operator()(**edge e1**,**edge e2**)**

**{**

**if** **(**e1**.**weight**>**e2**.**weight**)** **return** **true;**

**else** **return** **false;**

**}**

**};**

int main**()**

**{**

int n**,**start\_node**,**end\_node**;**

cin**>>**n**>>**start\_node**>>**end\_node**;**

vector**<**edge**>** e**;**

edge tem**;**

vector**<**edge**>** e\_v**[**n**];**

**while** **(**cin**>>**tem**.**start**>>**tem**.**end**>>**tem**.**weight**)**

**{**

e**.**push\_back**(**tem**);**

e\_v**[**tem**.**start**].**push\_back**(**tem**);**

**}**

node v**[**n**];**

int tmp**;**

**for** **(**int i**=**0**;**i**<**n**;**i**++)**

**{**

v**[**i**].**self**=**i**;**

v**[**i**].**parent**=**i**;**

v**[**i**].**rank**=**0**;**

v**[**i**].**in**=**0**;**

v**[**i**].**used**=false;**

v**[**i**].**reached**=false;**

**if** **(**i**!=**start\_node**)**

**{**

v**[**i**].**dist**=-**1**;**

v**[**i**].**prev**=NULL;**

**}**

**else**

**{**

v**[**i**].**dist**=**0**;**

v**[**i**].**prev**=&**v**[**i**];**

**}**

**}**

int p**,**q**;**

p**=**start\_node**;**

q**=**0**;**

v**[**p**].**reached**=true;**

**while** **(**q**<**n**)**

**{**

**for** **(**int i**=**0**;**i**<**n**;**i**++)**

**{**

**if** **(**v**[**p**].**reached **&&** **!**v**[**i**].**reached **&&** v**[**i**].**dist**!=-**1**)** p**=**i**;**

**if** **((!**v**[**i**].**reached**)&&(**v**[**i**].**dist**<**v**[**p**].**dist**)&&(**v**[**i**].**dist**!=-**1**))** p**=**i**;**

**}**

v**[**p**].**reached**=true;**

**if** **(**p**==**end\_node**)** **break;**

**for** **(**auto it**=**e\_v**[**p**].**begin**();**it**!=**e\_v**[**p**].**end**();**it**++)**

**{**

**if** **(**v**[**it**->**end**].**dist**==-**1**)**

**{**

v**[**it**->**end**].**dist**=**v**[**p**].**dist**+**it**->**weight**;** **}**

**if** **((**v**[**it**->**end**].**dist**!=-**1**)&&(**v**[**it**->**end**].**dist**>**v**[**p**].**dist**+**it**->**weight**))**

**{**

v**[**it**->**end**].**dist**=**v**[**p**].**dist**+**it**->**weight**;** **}**

**}**

q**++;**

**}**

**if** **(**v**[**end\_node**].**dist**==-**1**)** cout**<<**"No path exists!\n"**;**

**else** cout**<<**"Shortest path length is "**<<**v**[**end\_node**].**dist**<<**endl**;**

**for** **(**auto it**=**e**.**begin**();**it**!=**e**.**end**();**it**++)**

v**[**it**->**end**].**in**++;**

queue**<**int**>** Q**;**

**for** **(**int i**=**0**;**i**<**n**;**i**++)**

**if** **(**v**[**i**].**in**==**0**)**

Q**.**push**(**i**);**

**while** **(!**Q**.**empty**())**

**{**

tmp**=**Q**.**front**();**

Q**.**pop**();**

v**[**tmp**].**used**=true;**

**for** **(**auto it**=**e\_v**[**tmp**].**begin**();**it**!=**e\_v**[**tmp**].**end**();**it**++)**

**{**

v**[**it**->**end**].**in**--;**

**if** **(**v**[**it**->**end**].**in**==**0**)** Q**.**push**(**it**->**end**);**

**}**

**}**

bool t**=true;**

**for** **(**int i**=**0**;**i**<**n**;**i**++)**

**if** **(!**v**[**i**].**used**)** **{**t**=false;break;}**

**if** **(**t**)** cout**<<**"The graph is a DAG\n"**;**

**else** cout**<<**"The graph is not a DAG\n"**;**

priority\_queue**<**edge**,** vector**<**edge**>,** compare**>** E**;**

**for** **(**int i**=**0**;**i**<**e**.**size**();**i**++)**

E**.**push**(**e**[**i**]);**

int T**=**0**;**

int total\_weight**=**0**;**

int x**,**y**;**

**while** **(!**E**.**empty**())**

**{**

tem**=**E**.**top**();**

E**.**pop**();**

x**=**tem**.**start**;**

**while** **(**v**[**x**].**parent**!=**x**)**

**{**

x**=**v**[**x**].**parent**;**

**}**

x**=**v**[**x**].**parent**;**

y**=**tem**.**end**;**

**while** **(**v**[**y**].**parent**!=**y**)**

**{**

y**=**v**[**y**].**parent**;**

**}**

y**=**v**[**y**].**parent**;**

**if** **(**x**!=**y**)**

**{**

T**++;**

total\_weight**=**total\_weight**+**tem**.**weight**;**

**if** **(**v**[**x**].**rank**>**v**[**y**].**rank**)** v**[**y**].**parent**=**x**;**

**else** v**[**x**].**parent**=**y**;**

**if** **(**v**[**x**].**rank**==**v**[**y**].**rank**)** v**[**y**].**rank**++;**

**}**

**}**

**if** **(**T**==**n**-**1**)** cout**<<**"The total weight of MST is "**<<**total\_weight**<<**endl**;**

**else** cout**<<**"No MST exists!\n"**;**

**}**