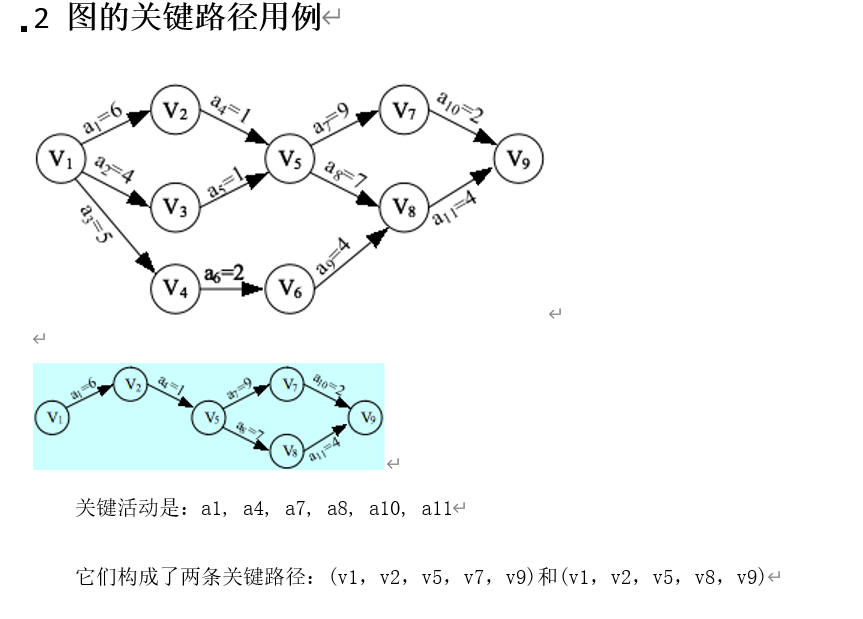
题目：求关键路径



代码：

#include <stdio.h>

#include <stdlib.h>

typedef struct NODE

{

int index;//入度

int data;//顶点

struct NODE\* next;

} NODE;

typedef struct node

{

int vex;//顶点

int dut;//边长

struct node\* next;

} node;

void CPM(NODE g[], int n, FILE\* fp);

int main()

{

int m, n;//顶点数，边数

int i, j, k, d;

node\* p;

FILE\* fp = fopen("input.txt", "r");

fscanf(fp, "%d %d", &m, &n);

NODE g[m + 1];

//结构体数组

for (j = 1; j <= m; j++)

{

g[j].next = NULL;

g[j].index = 0;

g[j].data = j;

}

//链表

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

{

fscanf(fp, "%d %d %d", &k, &j, &d);

node\* new = (node\*)malloc(sizeof(node));

new->next = g[k].next;

new->vex = j;

new->dut = d;

g[k].next = new;

}

//计算入度

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

{

p = g[i].next;

while (p != NULL)

{

g[p->vex].index++;

p = p->next;

}

}

fclose(fp);

FILE\* fp1 = fopen("result.txt", "w+");

CPM(g, m, fp1);

fclose(fp1);

}

void CPM(NODE g[], int n, FILE\* fp)

{

int ve[n + 1];

int vl[n + 1];

node\* p;

int i, j, k, m = 0;

int ftop = 0, btop = 0;//拓扑栈，逆拓扑栈

for (i = 1; i <= n; i++)

{

//入度为0的结点入栈

ve[i] = 0;

if (g[i].index == 0)

{

g[i].index = ftop;

ftop = i;

}

}

//求ve

while (ftop != 0)

{

j = ftop;

ftop = g[ftop].index;//退栈

g[j].index = btop;

btop = j;

m++;//顶点计数器

p = g[j].next;

//节点入度减一，计算ve

while (p != NULL)

{

k = p->vex;

g[k].index--;

if (g[k].index == 0)

{

g[k].index = ftop;

ftop = k;

}

if (ve[k] < ve[j] + p->dut)

ve[k] = ve[j] + p->dut;

p = p->next;

}

}

if (m < n)

{

printf("存在环\n");

return;

}

//求vl

for (i = 1; i <= n; i++)

{

vl[i] = ve[btop];

}

while (btop != 0)

{

j = btop;

btop = g[btop].index;//退栈

p = g[j].next;

while (p != NULL)

{

k = p->vex;

if (vl[j] > vl[k] - p->dut)

vl[j] = vl[k] - p->dut;

p = p->next;

}

}

//输出

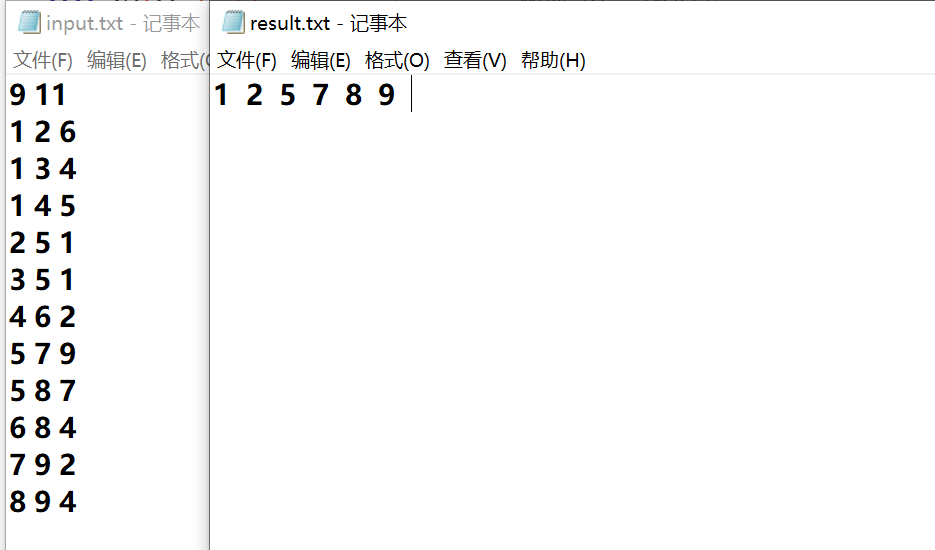
for (i = 1; i <= n; i++)

if (vl[i] == ve[i])

fprintf(fp, "%d ", g[i].data);

}

结果：



总结：求得事件的最早发生时间ve(j)和最迟发生时间vl(j)。如果活动ai由弧<j,k>表示，其持续时间记为dut(<j,k>)，则有如下关系：

e(i) = ve(j)

l(i) = vl(k) - dut(<j,k>)

求解ve(j)和vl(j)需分两个步进行：

1） 从ve(0)=0开始向前推进求得ve(j)

Ve(j) = Max{ve(i) + dut(<i,j>) };<i,j>属于T，j=1,2...,n-1

其中T是所有以第j个顶点为头的弧的集合。

2） 从vl(n-1) = ve(n-1)起向后推进求得vl(j)

vl(i) = Min{vl(j) - dut(<i,j>};<i,j>属于S，i=n-2,...,0

其中，S是所有以第i个顶点为尾的弧的集合。

这两个递推公式的计算必须分别在拓扑有序和逆拓扑有序的前提先进行。也就是说，ve(j-1)必须在vj的所有前驱的最早发生时间求得之后才能确定，而vl(j-1)必须在Vj的所有后继的最迟发生时间求得之后才能确定。因此可以在拓扑排序的基础上计算ve(j-1)和vl(j-1)。

入度域改造成拓扑栈和逆拓扑栈较难理解，因为入度域减为零之后这一项就没用了，所以当作拓扑栈和逆拓扑栈来节省空间。