实验九 图和二叉检索树

一、 实验目的

1. 熟悉图的最小生成树的概念，了解 Prim 算法和 Kruskal 算法。

2. 熟悉图的拓扑排序的概念

3. 熟悉图的关键路径和单源最短路径

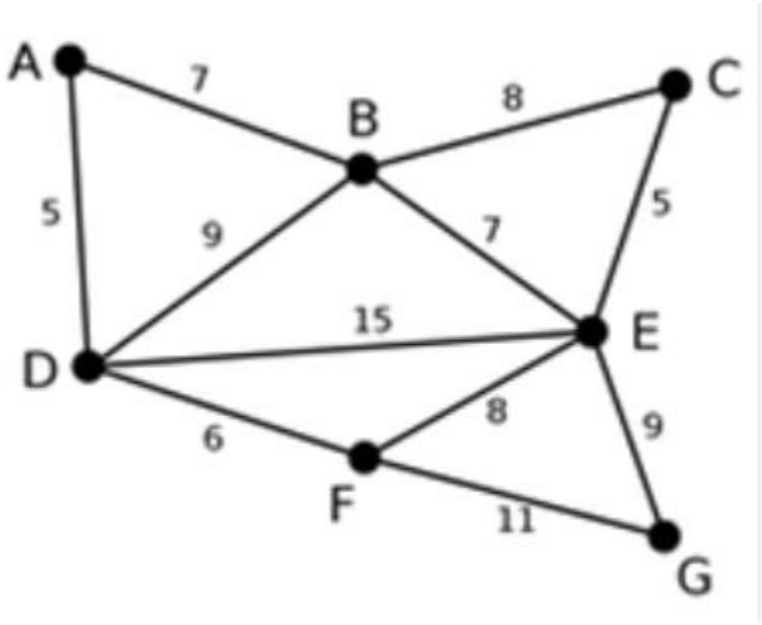
4. 熟悉二叉检索树

二、实验内容

1、实现 prim 和 Kruskal 算法

void prim(MGraphG,int v0,int adjxex[]);

测试数据：从 A 点出发，结果为：AD->DF->AB->BE->EC->EG



void Kruskal (MGraphG);

结果为：

（A,D）5

（C,E）5

（D,F）6

（A,B）7

（B,E）7

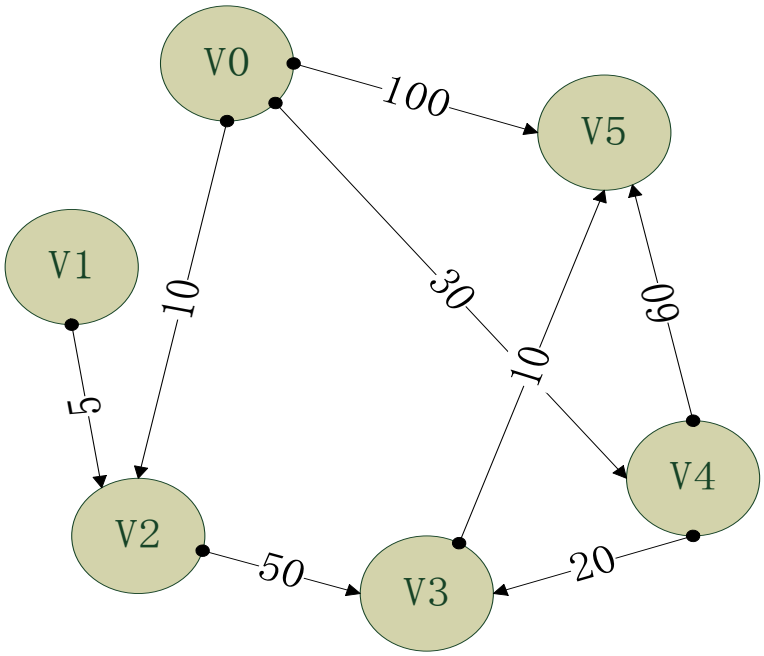
（E,G）9

2、 实现单源最短路径

voidDijkstra(MGraphG,int v0,int D[MAX_VERTEX_NUM],int P[MAX_VERTEX_NUM]);

测试

输入：



参考结果：

V0->V1：15　V0->V2->V1

V0->V2：10　V0->V2

V0->V3：50　V0->V4->V3

V0->V4：30　V0->V4

V0->V5：60　V0->V4->V3->V5

3、（附加题）二叉检索树的基本操作

二叉检索树是满足以下条件的二叉树：1.左子树上的所有节点值均小于根节点值，2 右子树上的所有节点值均不小于根节点值，3，左右子树也满足上述两个条件。

```
typedef struct BiTNode{
ElemType data;
struct BiTNode *lChild ,*rChild;
struct BiTNode *parent;
}BiTNode,*BiTree;
BiTree createBiTree(BiTree &T);   //创建二叉树
int BinearySearchTreeMax(BiTree T); //二叉检索树的最大值
int BinearySearchTreeMin(BiTree T);  //二叉检索树的最小值
void insert(BiTree &T, ElemType e);   //二叉检索树插入特定节点
void delete(BiTree &T, ElemType e);//二叉检索树删除特定节点
```

测试数据:5 3 1 # # 4 ## 7 6 # #8 # # ，其中 # 表示 NULL，可设为其他值，比如 0

即测试数据：5 3 1 0 0 4 0 0 7 6 0 0 8 0 0 （当数据类型定义成整型，可采用类似发方法处理）