考试说明：

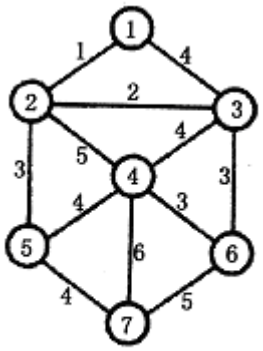
程序代码必须使用 C/C++语言完成，不接受任何其他编程语言。

以下有三道大题，第一题是必做题，剩下两题任选其一完成考试。

第一题总分 50 分，第二题总分 50 分，第三题总分 50 分。

不鼓励同时完成三道大题，否则在后两题中取较低分数统计总分，请特别注意。

一、编写代码利用 Kruscal 算法求解下图的最小代价生成树。

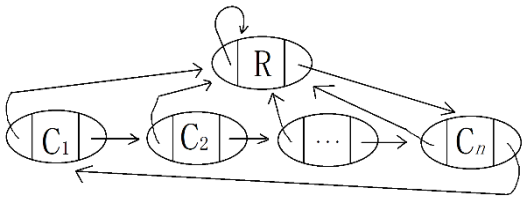


具体要求如下：

(1) 将以上无向图用邻接表或者邻接矩阵的形式表示出来；（5 分）

(2) 并不需要将所有的边按照权值大小进行排序，而是在算法的进行过程当中每次挑选一条当前权值最小的边即可。因此，在初始化的时候，要将所有的边按权值大小建堆（heap），之后每次取出堆顶元素，都要立即调整使之恢复成堆。建堆和后续的取堆顶元素的操作都会依赖“筛选”（或称“下推”）操作。该操作的函数原型类似：void adjustDown(EdgeType edge_set[],int pos1, int pos2);要求必须显式实现该函数；（15 分）

(3) 在向生成树中加边时，为了判断是否产生回路，要求构造并查集，并查集用数组中的双亲表示法作为存储结构。为了加快查询速度，并查集中每棵子树的结构如下图所示：



也就是说，树的高度是 2。树中每个结点除了包含数据域，还包含两个指针域。根结点的第一根指针指向自己，以表明自己是根结点。根结点的第二根指针指向任意一个孩子结点，如果不存在孩子结点则为空。孩子结点的第一根指针指向树的根结点，第二根指针指向根结点的下一个孩子。所有的孩子结点形成一个单向循环链表。注意，这里所说的指针在实现时其实是整数，代表某个元素在数组中的下标。

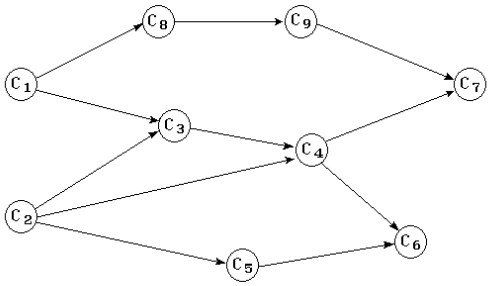
两棵子树合并之后，形成的子树依然保持上述的结构，要求将规模较小的子树合并进规模较大的子树。要求显式实现合并和查询的操作；（15 分）

(4) 按照边加入生成树的顺序输出每一条边，并输出最终的最小生成树的总代价；（5 分）

(5) 编写任意其他必要的函数，不许将所有代码都写在 main()函数中，并合理组织头文件和源文件；（5 分）

(6) 合理使用注释，代码拥有良好的风格。（5 分）

二、输入一个有向图，计算并输出该有向图的一个拓扑序列。如果该有向图没有拓扑序列，也要输出“没有拓扑序列”的提示。



具体要求如下：

(1) 将以上有向图用邻接矩阵表示出来，存放在数组 int graph[9][9]之中；（5 分）

(2) 采用深度优先遍历（DFS）的思想来构造拓扑序列，而且算法不表现为递归函数，而是用自建的数据结构（栈或队列）来存放临时数据；（5 分）

(3) 选择 C_1 为出发顶点，进行单趟 DFS，每个顶点在退栈时输出。如果经过一次 DFS 之后还有顶点没有被访问，重复以上操作，直到所有的顶点都被访问并输出，就会得到一个逆拓扑序列，将逆拓扑序列反向输出就得到拓扑序列；（15 分）

(4) 严格按照上述要求生成的拓扑序列是唯一的，若有错将扣分；（3 分）

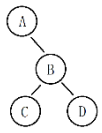
(5) 在图中增加一条边<C_6,C_2>，则算法必须发现回路并提示不能生成拓扑序列；（5 分）

(6) 编写任意其他必要的函数，不许将所有代码都写在 main()函数中，并合理组织头文件和源文件；（6 分）

(7) 深度优先遍历函数的原型类似下述形式：bool DFS(int idx, int path[], int &d); 期中 idx 为顶点编号，path 用来存放逆拓扑序列，d 为 path 中下一个插入数据的下标。DFS()函数返回 false 表明发现回路，不能生成拓扑序列。要求数组 path 的空间是在 DFS()之外动态申请和释放的；（6 分）

(8) 算法的时间复杂度低，代码拥有良好的风格。（5 分）

三、二叉树可以用带括号的广义表来表示，如下图所示的二叉树可以表示为：A(,B(C, D))。



若对此二叉树进行自底向上（bottom-up）的按层次遍历，得到输出序列：CDBA。

请按要求完成下述两项任务。

(a) 假设某二叉树可以表示成：A(B(C,D(E,F)),G(,H(I,)))。请设计一个算法，将此表达式作为输入，将其转换成为用二叉链表作为存储结构的二叉树。并输出这棵树的中序遍历次序。构建二叉树的算法可以为递归算法，也可以为非递归算法。

(b) 设计一个非递归算法，输出此二叉树的自底向上的按层次遍历序列。

评分标准如下：

(1) 必须清晰定义二叉树结点（Binary Tree Node）和二叉树（Binary Tree）的数据类型，并定义这些数据类型上的基本操作；（7 分）

(2) 能够正确地动态申请空间和释放空间；（7 分）

(3) 如果使用非递归算法，无论使用队列或栈作为辅助数据结构，要使用尽可能少的辅助空间完成任务；如果是使用递归算法，则要要用尽可能少的递归次数完成任务；（6 分）

(4) 能够合理地组织工程项目的结构和代码结构。整个项目要由头文件和 C/C++源文件共同

组成，不能将所有代码写在一个源文件中，也不允许在单个源文件中将所有代码写入一个函数；（5 分）

(5) 能够合理使用注释，代码拥有良好的风格；（5 分）

(6) 输出的中序遍历序列正确无误。（8 分）

(7) 能够正确输出二叉树的自底向上的按层次遍历序列，如果该项任务采用递归算法，则该项任务不得分。（12 分）