本次实验中，主要实现了图类和相关功能，图用邻接矩阵的方式存储，主要功能有判断图是不是有向图、无向图、有向无环图、连通图，求得图的拓朴排序结果，以及得到以点 1 为起点，到其他所有点的单源最短路径。

程序使用命令行参数输入，输出结果将被保存到与 main.exe 同目录下的 result.txt 中，输入格式是：先输入两个 int 型变量，分别代表结点数和边数，接下来 m 行数据均有 3 个 int 型变量，分别是边的起点、终点和边的权值。例如：

```
PS D:\desktop\data_structure\homework4\code\build> ./main.exe 5 6 5 1 4 5 4 5 5 2 3 1 2 2 4 2 6 2 3 7
```

就代表一共有 5 个点、6 条边，6 后面的数字每三个就代表了一条边的数据

接下来简单介绍程序的实现方法

下图为图类的基本结构，其中,printmartix 输出了图的邻接矩阵，isInDriected 和 isDriected 分别判断图是不是无向图和有向图，isDAG 判断图是不是有向无环图，isConnect 判断图是不是连通图，DFS 和 hasCycleDFS 是用于求是否是有向无环图和连通图的辅助搜索函数，topologicalSort 是求拓朴排序的函数，dijkstra 是求点 node 的单源最短函数，这里如果没有的话将输出“”

```
class graph
{
public:
    int nsum;//结点数量
    int **matrix;
    int *appear;
    graph(int n):nsum(n)···
    void printmatrix();
    bool isInDriected();
    bool isDriected();
    bool isDAG();
    bool isConnect();
    void DFS(int n);
    bool hasCycleDFS(int node, int *instack);
    int* topologicalSort();
    void dijkstra(int node);
};
```

以下为一个输出结果，图中含有 5 个点，6 条边，输出数据中有它的邻接矩阵，这个图是有向图，是有向无环图，但不是连通图，拓朴排序的结果是 5 1 4 2 3，点 1 到 2 和 3 的最短路径分别是 2 和 9，无法到达 4 和 5，因此不存在最短路径

实现思路：

判断是否是有向图和无向图时，遍历整个邻接矩阵，如果对任意的 i，j，都有 i 到 j 的边权值等于 j 到 i 的边权值，就是无向图，如果存在一对 ij 使得 i 到 j 的边权值不为 MAXINT（也就是存在一条边）且 i 到 j 的边权值≠j 到 i 的边权值，就是有向图。

判断是否是有向无环图的时候，先判断是否是无向图，若是就直接 return false，否则开始依次遍历每个点，通过 hasCycleDFS 和栈，每次把当前结点所有有边且未访问的邻居结点入栈，如果栈中已存在该结点，就说明不是有向无环图。

判断是否是连通图时， 从结点 1 开始进行依次深度优先搜索，然后检查所有结点，如果存在未被访问的结点，就说明图有不止一个连通分量，也就不是连通图。

求拓扑排序时，先计算所有结点的入度，然后依次寻找入度为 0 的点，存到结果数组并删除该结点和与该

结点关联的边，并把与这些边关联的结点入度-1，最终把结果数组翻转即为拓扑排序结果。
求单源最短路径即为 dijkstra 算法，简单默写代码即结束。