# 1、解题思路

1.1、需求分析：

利用warshall算法求传递闭包

1.2、实验原理：  
（1）闭包:

定义4.17 设R是非空集合A上的关系, R 的自反 (对称或传递) 闭包 是A上的关系R′ , 使得 R′满足以下条件： (1) R′是自反的（对称的或传递的） (2) R ⊆R′ (3) 对A上任何包含R 的自反（对称或传递）关系R′′ 有 R′⊆R′′. 一般将 R 的自反闭包记作 r(R), 对称闭包记作 s(R), 传递 闭包记作 t(R)。

### （2）Warshall算法原理：

按照定义，t(R)=R∪∪∪…。从图的角度理解，若在原图的基础上，在n步之内，两个点之间互相可达，则这两个点之间的关系应该被包括在传递闭包内。

假设要判断顶点到是否可达。如果从到无弧，则从到不存在一条长度为1的路径，但不能断定到不存在长度不为1的路径，尚需进行n次试探。首先考虑路径到是否存在。如果存在，则再查看到之间是否存在通路。假如存在，则到存在通路。假如不存在，再增加一个顶点进行试探，依次类推。在一般情况下，代表中间存在t-s个节点时到可能存在的通路。这样,在经过n次试探后，最后必能判断出到之间是否存在通路。按此方法,可以判断各对顶点间是否存在通路。

## 1.3、设计思路：

传递闭包的一种有效算法—Warshall算法，这种算法也便于计算机实现。   
(1)置新矩阵A＝M；   
(2)i＝1；   
(3)对所有j如果A[j，i]＝1，则对k＝1，2，…，n，A[j，k]＝A[j，k]∨A[i，k]；   
(4)i加1；（i是行，j是列）   
(5)如果i≤n，则转到步骤3)，否则停止。

# 2、数据结构：

在该实验中，主要使用数组：

由于在计算机中，二维数组实际上占用一段连续的内存，所以，可以使用M[n\*n]的一维数组来模拟二维数组M[n][n]。使用一维数组的好处是更方便进行内存动态管理，节约空间。

# 3、核心算法：

for (int k = 0; k < n; ++k)

{

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

for (int j = 0; j < n; ++j)

if (M[i \* n + k] && M[k \* n + j])M[i \* n + j] = 1;

}

# 4、程序测试：

## a、基本测试：

1、测试输入：

3

0 0 1

1 0 1

0 1 1

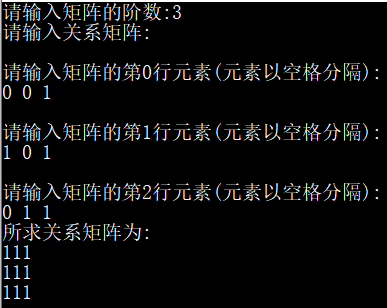
预期结果：

111

111

111

实验结果：



2、测试输入：

4

0 1 0 0

0 0 0 1

0 0 0 0

1 0 1 0

预期：

1111

1111

0000

1111

实验结果：

