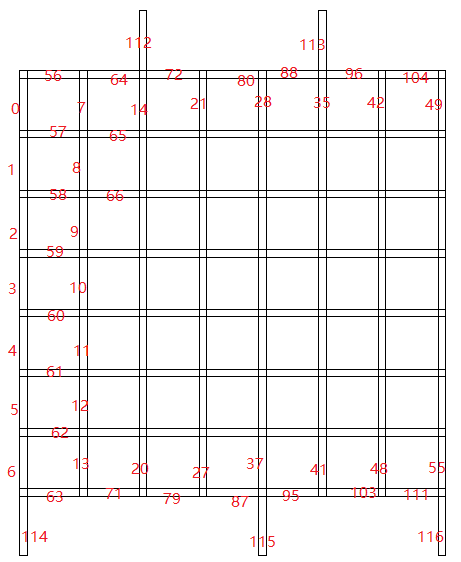
主要接口 vector<double> caluconspeed(int num, vector<double>&length, int i1, int i2, int o1, int o2, int o3)

* num: 正方形网格的边长（即网格一行的节点数量，比如8X8的网格，一行有8个节点）
* length: 存储网格各个边的边长的数组，边的编号方式在之后介绍。Length.size()应当等于2\*num\*num-2\*num+5。前2\*num\*num-2\*num个边对应内部的边。后5条边代表两个输出管道与三个输出管道的长度。对于不存在的边，在数组中的值设为0。
* i1,i2: 两个输入管道分别在第几列（0<=i1<i2<num）
* o1,o2,o3: 三个输出管道分别在第几列（0<=o1<o2<o3<num）
* 返回值：按照o1，o2，o3的顺序表示三个输出管道的流速

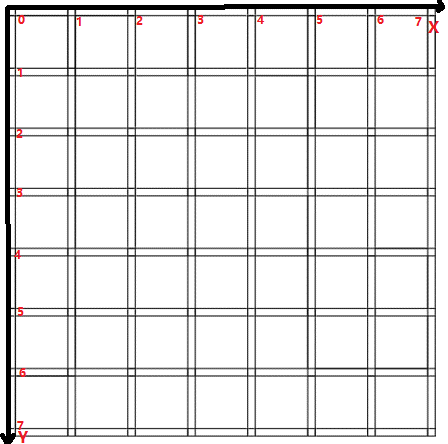
程序主要流程：

* 确定节点与管道的连接关系。
* 读入所有管道的长度。
* 准备构建方程组：
  + 不存在的管道，流速为0，构建方程。
  + 对于任意节点，流入流速和等于流出流速和，构建方程。
  + 对于芯片结构的所有最小环路确定电势差为0，构建方程。
  + 任意两个输出端口之间的电势差为0，构建两个方程。
  + 根据两个输入端口的输入流速，构建两个方程。
* 通过行列式的方式，计算方程组。

芯片管道的编号方式为：对于不包含输入与输出管道的网格结构，首先从左到右，从上到下，对于竖直的管道从0开始编号直到n\*n-n-1；之后从左到右，从上到下，对于水平的管道从n\*n-n开始编号直到2\*n\*n-2\*n-1。对于i1,i2,o1,o2,o3分别编号为2\*n\*n-2\*n，2\*n\*n-2\*n+1，2\*n\*n-2\*n+2，2\*n\*n-2\*n+3，2\*n\*n-2\*n+4。管道的总数为2\*n\*n-2\*n+5。如下图为一个8X8网络中的编号方式。



节点的二维表示：程序中对于每个节点以二维形式(x,y)表示，x代表节点处于第几列，y代表节点处于第几行。下图为8X8网络中的节点编号样例。



提示：由于本算法是以方程的形式求解，由于所有的管道长度相同，对于基础要求可以输入所有的管道长度都是1。

之后同学提出的一些问题，会在<https://github.com/darknessnone/2018-summerwork/wiki/Q&A>更新。