数据1------------------------------------------------

int layerNum = 1;  
int netsNum = 2;  
//int obstaclesNum = 19;  
int gridXmax = 6;  
int gridYmax = 4;//初始化数据  
int sum = 0;  
GraphModel graphModel[] = new GraphModel[layerNum];  
for(int i =0;i<layerNum;i++){  
 graphModel[i] = new GraphModel(gridXmax,gridYmax);  
}  
for(int i = 0;i<gridXmax;i++){  
 for(int j = 0;j<gridYmax;j++) {  
 //double res = Math.random();  
 /\*if(j == 5){  
 graphModel[0].graph[i][j] = 1;  
 }  
 else{  
 graphModel[0].graph[i][j] = 0;  
 }\*/  
 graphModel[0].graph[i][j] = 0;  
 sum = sum + graphModel[0].graph[i][j];  
 }  
}  
Coordinate sources[] = new Coordinate[netsNum];//起点定义  
sources[0] =new Coordinate(0,2,gridXmax,gridYmax);  
sources[1] =new Coordinate(5,2,gridXmax,gridYmax);  
/\*sources[2] =new Coordinate(0,7,gridXmax,gridYmax);  
sources[3] =new Coordinate(10,7,gridXmax,gridYmax);\*/  
for(int i = 0; i<sources.length;i++){  
 graphModel[0].graph[sources[i].x][sources[i].y] = 1;  
}  
//初始化，后续更改为更灵活的方式  
Coordinate sinks[] = new Coordinate[netsNum];//终点定义  
sinks[0] =new Coordinate(2,0,gridXmax,gridYmax);  
sinks[1] =new Coordinate(3,0,gridXmax,gridYmax);  
/\*sinks[2] =new Coordinate(10,1,gridXmax,gridYmax);  
sinks[3] =new Coordinate(0,1,gridXmax,gridYmax);\*/  
  
NetPath path[] = new NetPath[netsNum];//net定义，包含目标长度，已走长度，未走长度，以及路径坐标点列表  
int distance[] = new int[netsNum];//初始距离  
int maxDistance = 0;  
for(int i = 0;i<netsNum;i++) {  
 distance[i] = Math.*abs*(sinks[i].x - sources[i].x) + Math.*abs*(sinks[i].y - sources[i].y);  
 if(distance[i]> maxDistance){  
 maxDistance = distance[i];  
 }  
}  
//System.out.println("目标距离"+maxDistance);//测试是否生效  
/\*path[0] = new NetPath(maxDistance,maxDistance,0);  
path[1] = new NetPath(maxDistance,maxDistance,0);  
path[2] = new NetPath(maxDistance,maxDistance,0);  
path[3] = new NetPath(maxDistance,maxDistance,0);\*/  
for(int i = 0; i<path.length;i++){  
 path[i] = new NetPath(maxDistance,maxDistance,0);  
 Coordinate temp = new Coordinate(sources[i].x,sources[i].y,gridXmax,gridYmax);  
 path[i].routingPath.add(temp);  
}  
//初始化  
SymmetricNets syNetPairs[] = new SymmetricNets[1];  
syNetPairs[0] = new SymmetricNets(0,1);//对称nets定义  
//syNetPairs[1] = new SymmetricNets(0,1);  
//System.out.println(syNetPairs1.isIntersected());//判断是否有intersection  
//System.out.println(syNetPairs2.isIntersected());  
int sourcesOrder[] = {0,1};//依次为y轴，x轴，x值最大那条线上的点，y值最大那条线上的点，如何用程序转换后续解决，考虑在Coordinate里增加一个参数记录其标号  
int sinksOrder[] = {0,1};

数据2------------------------------------------------

int layerNum = 2;  
int netsNum = 2;  
//int obstaclesNum = 19;  
int gridXmax = 5;  
int gridYmax = 5;//初始化数据  
int sum = 0;  
GraphModel graphModel[] = new GraphModel[layerNum];  
for(int i =0;i<layerNum;i++){  
 graphModel[i] = new GraphModel(gridXmax,gridYmax);  
}  
for(int i = 0;i<gridXmax;i++){  
 for(int j = 0;j<gridYmax;j++) {  
 //double res = Math.random();  
 /\*if(j == 5){  
 graphModel[0].graph[i][j] = 1;  
 }  
 else{  
 graphModel[0].graph[i][j] = 0;  
 }\*/  
 graphModel[0].graph[i][j] = 0;  
 sum = sum + graphModel[0].graph[i][j];  
 }  
}  
Coordinate sources[] = new Coordinate[netsNum];//起点定义  
  
sources[0] =new Coordinate(0,4,gridXmax,gridYmax);  
sources[1] =new Coordinate(4,4,gridXmax,gridYmax);  
/\*sources[2] =new Coordinate(0,7,gridXmax,gridYmax);  
sources[3] =new Coordinate(10,7,gridXmax,gridYmax);\*/  
for(int i = 0; i<sources.length;i++){  
 graphModel[0].graph[sources[i].x][sources[i].y] = 1;  
}  
//初始化，后续更改为更灵活的方式  
Coordinate sinks[] = new Coordinate[netsNum];//终点定义  
sinks[0] =new Coordinate(4,1,gridXmax,gridYmax);  
sinks[1] =new Coordinate(0,1,gridXmax,gridYmax);  
/\*sinks[2] =new Coordinate(10,1,gridXmax,gridYmax);  
sinks[3] =new Coordinate(0,1,gridXmax,gridYmax);\*/  
  
NetPath path[] = new NetPath[netsNum];//net定义，包含目标长度，已走长度，未走长度，以及路径坐标点列表  
int distance[] = new int[netsNum];//初始距离  
int maxDistance = 0;  
for(int i = 0;i<netsNum;i++) {  
 distance[i] = Math.*abs*(sinks[i].x - sources[i].x) + Math.*abs*(sinks[i].y - sources[i].y);  
 if(distance[i]> maxDistance){  
 maxDistance = distance[i];  
 }  
}  
//System.out.println("目标距离"+maxDistance);//测试是否生效  
/\*path[0] = new NetPath(maxDistance,maxDistance,0);  
path[1] = new NetPath(maxDistance,maxDistance,0);  
path[2] = new NetPath(maxDistance,maxDistance,0);  
path[3] = new NetPath(maxDistance,maxDistance,0);\*/  
for(int i = 0; i<path.length;i++){  
 path[i] = new NetPath(maxDistance,maxDistance,0);  
 Coordinate temp = new Coordinate(sources[i].x,sources[i].y,gridXmax,gridYmax);  
 path[i].routingPath.add(temp);  
}  
//初始化  
SymmetricNets syNetPairs[] = new SymmetricNets[1];  
syNetPairs[0] = new SymmetricNets(0,1);//对称nets定义  
//syNetPairs[1] = new SymmetricNets(0,1);  
//System.out.println(syNetPairs1.isIntersected());//判断是否有intersection  
//System.out.println(syNetPairs2.isIntersected());  
int sourcesOrder[] = {0,1};//依次为y轴，x轴，x值最大那条线上的点，y值最大那条线上的点(不一定，总之环绕图边一周），如何用程序转换后续解决，考虑在Coordinate里增加一个参数记录其标号  
int sinksOrder[] = {1,0};  
int lcs[][] = new int[netsNum][netsNum];//用于分层，后续修改为更灵活方式

数据3------------------------------------------------

int layerNum = 3;  
int netsNum = 4;  
//int obstaclesNum = 19;  
int gridXmax = 11;  
int gridYmax = 9;//初始化数据  
int sum = 0;  
GraphModel graphModel[] = new GraphModel[layerNum];  
for(int i =0;i<layerNum;i++){  
 graphModel[i] = new GraphModel(gridXmax,gridYmax);  
}  
for(int i = 0;i<gridXmax;i++){  
 for(int j = 0;j<gridYmax;j++) {  
 //double res = Math.random();  
 if(j == 5){  
 graphModel[0].graph[i][j] = 1;  
 }  
 else{  
 graphModel[0].graph[i][j] = 0;  
 }  
 //graphModel[0].graph[i][j] = 0;  
 sum = sum + graphModel[0].graph[i][j];  
 }  
}  
Coordinate sources[] = new Coordinate[netsNum];//起点定义  
sources[0] =new Coordinate(0,8,gridXmax,gridYmax);  
sources[1] =new Coordinate(10,8,gridXmax,gridYmax);  
sources[2] =new Coordinate(0,7,gridXmax,gridYmax);  
sources[3] =new Coordinate(10,7,gridXmax,gridYmax);  
for(int i = 0; i<sources.length;i++){  
 graphModel[0].graph[sources[i].x][sources[i].y] = 1;  
}  
//初始化，后续更改为更灵活的方式  
Coordinate sinks[] = new Coordinate[netsNum];//终点定义  
sinks[0] =new Coordinate(10,4,gridXmax,gridYmax);  
sinks[1] =new Coordinate(0,4,gridXmax,gridYmax);  
sinks[2] =new Coordinate(10,1,gridXmax,gridYmax);  
sinks[3] =new Coordinate(0,1,gridXmax,gridYmax);  
  
NetPath path[] = new NetPath[netsNum];//net定义，包含目标长度，已走长度，未走长度，以及路径坐标点列表  
int distance[] = new int[netsNum];//初始距离  
int maxDistance = 0;  
for(int i = 0;i<netsNum;i++) {  
 distance[i] = Math.*abs*(sinks[i].x - sources[i].x) + Math.*abs*(sinks[i].y - sources[i].y);  
 if(distance[i]> maxDistance){  
 maxDistance = distance[i];  
 }  
}  
//System.out.println("目标距离"+maxDistance);//测试是否生效  
for(int i = 0; i<path.length;i++){  
 path[i] = new NetPath(maxDistance,maxDistance,0);  
 Coordinate temp = new Coordinate(sources[i].x,sources[i].y,gridXmax,gridYmax);  
 path[i].routingPath.add(temp);  
}  
//初始化  
SymmetricNets syNetPairs[] = new SymmetricNets[2];  
syNetPairs[0] = new SymmetricNets(0,1);//对称nets定义  
syNetPairs[1] = new SymmetricNets(2,3);  
//System.out.println(syNetPairs1.isIntersected());//判断是否有intersection  
//System.out.println(syNetPairs2.isIntersected());  
int sourcesOrder[] = {0,2,3,1};//依次为y轴，x轴，x值最大那条线上的点，y值最大那条线上的点，如何用程序转换后续解决，考虑在Coordinate里增加一个参数记录其标号  
int sinksOrder[] = {3,1,0,2};

数据4------------------------------------------------

int layerNum = 4;  
int netsNum = 10;  
//int obstaclesNum = 19;  
int gridXmax = 15;  
int gridYmax = 10;//初始化数据  
int sum = 0;  
GraphModel graphModel[] = new GraphModel[layerNum];  
for(int i =0;i<layerNum;i++){  
 graphModel[i] = new GraphModel(gridXmax,gridYmax);  
}  
/\*for(int k =0;k<layerNum;k++){//在各层随机生成障碍  
 for(int i = 0;i<gridXmax;i++){  
 for(int j = 0;j<gridYmax;j++) {  
 double res = Math.random();  
 if(res>0.05){  
 graphModel[k].graph[i][j] = 0;  
 }  
 else{  
 graphModel[k].graph[i][j] = 1;  
 }  
 //graphModel[0].graph[i][j] = 0;  
 sum = sum + graphModel[k].graph[i][j];  
 }  
 }  
}\*/  
for(int i = 0;i<gridXmax;i++){  
 for(int j = 0;j<gridYmax;j++) {  
 /\*double res = Math.random();  
 if(res>0.05){  
 graphModel[0].graph[i][j] = 0;  
 }  
 else{  
 graphModel[0].graph[i][j] = 1;  
 }\*/  
 graphModel[0].graph[i][j] = 0;  
 sum = sum + graphModel[0].graph[i][j];  
 }  
}  
  
Coordinate sources[] = new Coordinate[netsNum];//起点定义  
  
sources[0] =new Coordinate(0,1,gridXmax,gridYmax);  
sources[1] =new Coordinate(0,3,gridXmax,gridYmax);  
sources[2] =new Coordinate(0,5,gridXmax,gridYmax);  
sources[3] =new Coordinate(0,7,gridXmax,gridYmax);  
sources[4] =new Coordinate(0,9,gridXmax,gridYmax);  
sources[5] =new Coordinate(14,1,gridXmax,gridYmax);  
sources[6] =new Coordinate(14,3,gridXmax,gridYmax);  
sources[7] =new Coordinate(14,5,gridXmax,gridYmax);  
sources[8] =new Coordinate(14,7,gridXmax,gridYmax);  
sources[9] =new Coordinate(14,9,gridXmax,gridYmax);  
/\*sources[2] =new Coordinate(0,7,gridXmax,gridYmax);  
sources[3] =new Coordinate(10,7,gridXmax,gridYmax);\*/  
for(int i = 0; i<sources.length;i++){  
 graphModel[0].graph[sources[i].x][sources[i].y] = 1;  
}  
//初始化，后续更改为更灵活的方式  
Coordinate sinks[] = new Coordinate[netsNum];//终点定义  
sinks[0] =new Coordinate(14,6,gridXmax,gridYmax);  
sinks[1] =new Coordinate(14,8,gridXmax,gridYmax);  
sinks[2] =new Coordinate(10,0,gridXmax,gridYmax);  
sinks[3] =new Coordinate(11,0,gridXmax,gridYmax);  
sinks[4] =new Coordinate(12,0,gridXmax,gridYmax);  
sinks[5] =new Coordinate(0,6,gridXmax,gridYmax);  
sinks[6] =new Coordinate(0,8,gridXmax,gridYmax);  
sinks[7] =new Coordinate(4,0,gridXmax,gridYmax);  
sinks[8] =new Coordinate(3,0,gridXmax,gridYmax);  
sinks[9] =new Coordinate(2,0,gridXmax,gridYmax);  
/\*sinks[2] =new Coordinate(10,1,gridXmax,gridYmax);  
sinks[3] =new Coordinate(0,1,gridXmax,gridYmax);\*/  
  
NetPath path[] = new NetPath[netsNum];//net定义，包含目标长度，已走长度，未走长度，以及路径坐标点列表  
int distance[] = new int[netsNum];//初始距离  
int maxDistance = 0;  
for(int i = 0;i<netsNum;i++) {  
 distance[i] = Math.*abs*(sinks[i].x - sources[i].x) + Math.*abs*(sinks[i].y - sources[i].y);  
 if(distance[i]> maxDistance){  
 maxDistance = distance[i];  
 }  
}  
System.*out*.println("目标距离"+maxDistance);//测试是否生效  
/\*path[0] = new NetPath(maxDistance,maxDistance,0);  
path[1] = new NetPath(maxDistance,maxDistance,0);  
path[2] = new NetPath(maxDistance,maxDistance,0);  
path[3] = new NetPath(maxDistance,maxDistance,0);\*/  
for(int i = 0; i<path.length;i++){  
 path[i] = new NetPath(maxDistance,maxDistance,0);  
 Coordinate temp = new Coordinate(sources[i].x,sources[i].y,gridXmax,gridYmax);  
 path[i].routingPath.add(temp);  
}  
//初始化  
SymmetricNets syNetPairs[] = new SymmetricNets[5];  
syNetPairs[0] = new SymmetricNets(0,5);//对称nets定义  
syNetPairs[1] = new SymmetricNets(1,6);  
syNetPairs[2] = new SymmetricNets(2,7);  
syNetPairs[3] = new SymmetricNets(3,8);  
syNetPairs[4] = new SymmetricNets(4,9);  
//System.out.println(syNetPairs1.isIntersected());//判断是否有intersection  
//System.out.println(syNetPairs2.isIntersected());  
int sourcesOrder[] = {0,1,2,3,4,9,8,7,6,5};//依次为y轴，x轴，x值最大那条线上的点，y值最大那条线上的点(不一定，总之环绕图边一周），如何用程序转换后续解决，考虑在Coordinate里增加一个参数记录其标号  
int sinksOrder[] = {6,5,9,8,7,2,3,4,0,1};

数据5------------------------------------------------

int layerNum = 1;  
 int netsNum = 20;  
//int obstaclesNum = 19;  
 int gridXmax = 120;  
 int gridYmax = 120;//初始化数据  
 int sum = 0;  
 GraphModel graphModel[] = new GraphModel[layerNum];  
 for(int i =0;i<layerNum;i++){  
 graphModel[i] = new GraphModel(gridXmax,gridYmax);  
 }  
 for(int i = 0;i<gridXmax;i++){  
 for(int j = 0;j<gridYmax;j++) {  
 //double res = Math.random();  
 /\*if(j == 5){  
 graphModel[0].graph[i][j] = 1;  
 }  
 else{  
 graphModel[0].graph[i][j] = 0;  
 }\*/  
 graphModel[0].graph[i][j] = 0;  
 sum = sum + graphModel[0].graph[i][j];  
 }  
 }  
 Coordinate sources[] = new Coordinate[netsNum];//起点定义  
 for(int i= 0; i<netsNum;i++){  
 if(i<10){  
 sources[i] = new Coordinate(0,119-3\*i,gridXmax,gridYmax);  
 }  
 else{  
 sources[i] = new Coordinate(119,119-3\*(i-10),gridXmax,gridYmax);  
 }  
 }  
/\*sources[0] =new Coordinate(0,2,gridXmax,gridYmax);  
sources[1] =new Coordinate(5,2,gridXmax,gridYmax);\*/  
/\*sources[2] =new Coordinate(0,7,gridXmax,gridYmax);  
sources[3] =new Coordinate(10,7,gridXmax,gridYmax);\*/  
 for(int i = 0; i<sources.length;i++){  
 graphModel[0].graph[sources[i].x][sources[i].y] = 1;  
 }  
//初始化，后续更改为更灵活的方式  
 Coordinate sinks[] = new Coordinate[netsNum];//终点定义  
 for(int i= 0; i<netsNum;i++){  
 if(i<10){  
 sinks[i] = new Coordinate(59-3\*i,0,gridXmax,gridYmax);  
 }  
 else{  
 sinks[i] = new Coordinate(60+3\*(i-10),0,gridXmax,gridYmax);  
 }  
 }  
/\*sinks[0] =new Coordinate(2,0,gridXmax,gridYmax);  
sinks[1] =new Coordinate(3,0,gridXmax,gridYmax);\*/  
/\*sinks[2] =new Coordinate(10,1,gridXmax,gridYmax);  
sinks[3] =new Coordinate(0,1,gridXmax,gridYmax);\*/  
  
 NetPath path[] = new NetPath[netsNum];//net定义，包含目标长度，已走长度，未走长度，以及路径坐标点列表  
 int distance[] = new int[netsNum];//初始距离  
 int maxDistance = 0;  
 for(int i = 0;i<netsNum;i++) {  
 distance[i] = Math.*abs*(sinks[i].x - sources[i].x) + Math.*abs*(sinks[i].y - sources[i].y);  
 if(distance[i]> maxDistance){  
 maxDistance = distance[i];  
 }  
 }  
 System.*out*.println("目标距离"+maxDistance);//测试是否生效  
/\*path[0] = new NetPath(maxDistance,maxDistance,0);  
path[1] = new NetPath(maxDistance,maxDistance,0);  
path[2] = new NetPath(maxDistance,maxDistance,0);  
path[3] = new NetPath(maxDistance,maxDistance,0);\*/  
 for(int i = 0; i<path.length;i++){  
 path[i] = new NetPath(maxDistance,maxDistance,0);  
 Coordinate temp = new Coordinate(sources[i].x,sources[i].y,gridXmax,gridYmax);  
 path[i].routingPath.add(temp);  
 }  
//初始化  
 SymmetricNets syNetPairs[] = new SymmetricNets[10];  
 for(int i =0;i<10;i++){  
 syNetPairs[i] = new SymmetricNets(i,i+10);  
 }  
//syNetPairs[0] = new SymmetricNets(0,10);//对称nets定义  
//syNetPairs[1] = new SymmetricNets(0,1);  
//System.out.println(syNetPairs1.isIntersected());//判断是否有intersection  
//System.out.println(syNetPairs2.isIntersected());  
 int sourcesOrder[] = {9,8,7,6,5,4,3,2,1,0,10,11,12,13,14,15,16,17,18,19};//依次为y轴，x轴，x值最大那条线上的点，y值最大那条线上的点(不一定，总之环绕图边一周），如何用程序转换后续解决，考虑在Coordinate里增加一个参数记录其标号  
 int sinksOrder[] = {9,8,7,6,5,4,3,2,1,0,10,11,12,13,14,15,16,17,18,19};

数据6------------------------------------------------

int layerNum = 2;  
 int netsNum = 40;  
//int obstaclesNum = 19;  
 int gridXmax = 120;  
 int gridYmax = 120;//初始化数据  
 int sum = 0;  
 GraphModel graphModel[] = new GraphModel[layerNum];  
 for(int i =0;i<layerNum;i++){  
 graphModel[i] = new GraphModel(gridXmax,gridYmax);  
 }  
 for(int i = 0;i<gridXmax;i++){  
 for(int j = 0;j<gridYmax;j++) {  
 //double res = Math.random();  
 /\*if(j == 5){  
 graphModel[0].graph[i][j] = 1;  
 }  
 else{  
 graphModel[0].graph[i][j] = 0;  
 }\*/  
 graphModel[0].graph[i][j] = 0;  
 sum = sum + graphModel[0].graph[i][j];  
 }  
 }  
 Coordinate sources[] = new Coordinate[netsNum];//起点定义  
 for(int i= 0; i<netsNum;i++){  
 if(i<20){  
 sources[i] = new Coordinate(0,119-2\*i,gridXmax,gridYmax);  
 }  
 else{  
 sources[i] = new Coordinate(119,119-2\*(i-20),gridXmax,gridYmax);  
 }  
 }  
/\*sources[0] =new Coordinate(0,2,gridXmax,gridYmax);  
sources[1] =new Coordinate(5,2,gridXmax,gridYmax);\*/  
/\*sources[2] =new Coordinate(0,7,gridXmax,gridYmax);  
sources[3] =new Coordinate(10,7,gridXmax,gridYmax);\*/  
 for(int i = 0; i<sources.length;i++){  
 graphModel[0].graph[sources[i].x][sources[i].y] = 1;  
 }  
//初始化，后续更改为更灵活的方式  
 Coordinate sinks[] = new Coordinate[netsNum];//终点定义  
 for(int i= 0; i<netsNum;i++){  
 if(i<20){  
 sinks[i] = new Coordinate(119,40-2\*i,gridXmax,gridYmax);  
 }  
 else{  
 sinks[i] = new Coordinate(0,40-2\*(i-20),gridXmax,gridYmax);  
 }  
 }  
/\*sinks[0] =new Coordinate(2,0,gridXmax,gridYmax);  
sinks[1] =new Coordinate(3,0,gridXmax,gridYmax);\*/  
/\*sinks[2] =new Coordinate(10,1,gridXmax,gridYmax);  
sinks[3] =new Coordinate(0,1,gridXmax,gridYmax);\*/  
  
 NetPath path[] = new NetPath[netsNum];//net定义，包含目标长度，已走长度，未走长度，以及路径坐标点列表  
 int distance[] = new int[netsNum];//初始距离  
 int maxDistance = 0;  
 for(int i = 0;i<netsNum;i++) {  
 distance[i] = Math.*abs*(sinks[i].x - sources[i].x) + Math.*abs*(sinks[i].y - sources[i].y);  
 if(distance[i]> maxDistance){  
 maxDistance = distance[i];  
 }  
 }  
 System.*out*.println("目标距离"+maxDistance);//测试是否生效  
/\*path[0] = new NetPath(maxDistance,maxDistance,0);  
path[1] = new NetPath(maxDistance,maxDistance,0);  
path[2] = new NetPath(maxDistance,maxDistance,0);  
path[3] = new NetPath(maxDistance,maxDistance,0);\*/  
 for(int i = 0; i<path.length;i++){  
 path[i] = new NetPath(maxDistance,maxDistance,0);  
 Coordinate temp = new Coordinate(sources[i].x,sources[i].y,gridXmax,gridYmax);  
 path[i].routingPath.add(temp);  
 }  
//初始化  
 SymmetricNets syNetPairs[] = new SymmetricNets[10];  
 for(int i =0;i<10;i++){  
 syNetPairs[i] = new SymmetricNets(i,i+10);  
 }  
//syNetPairs[0] = new SymmetricNets(0,10);//对称nets定义  
//syNetPairs[1] = new SymmetricNets(0,1);  
//System.out.println(syNetPairs1.isIntersected());//判断是否有intersection  
//System.out.println(syNetPairs2.isIntersected());  
 int sourcesOrder[] = {19,18,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39};//依次为y轴，x轴，x值最大那条线上的点，y值最大那条线上的点(不一定，总之环绕图边一周），如何用程序转换后续解决，考虑在Coordinate里增加一个参数记录其标号  
 int sinksOrder[] = {20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,19,18,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0};