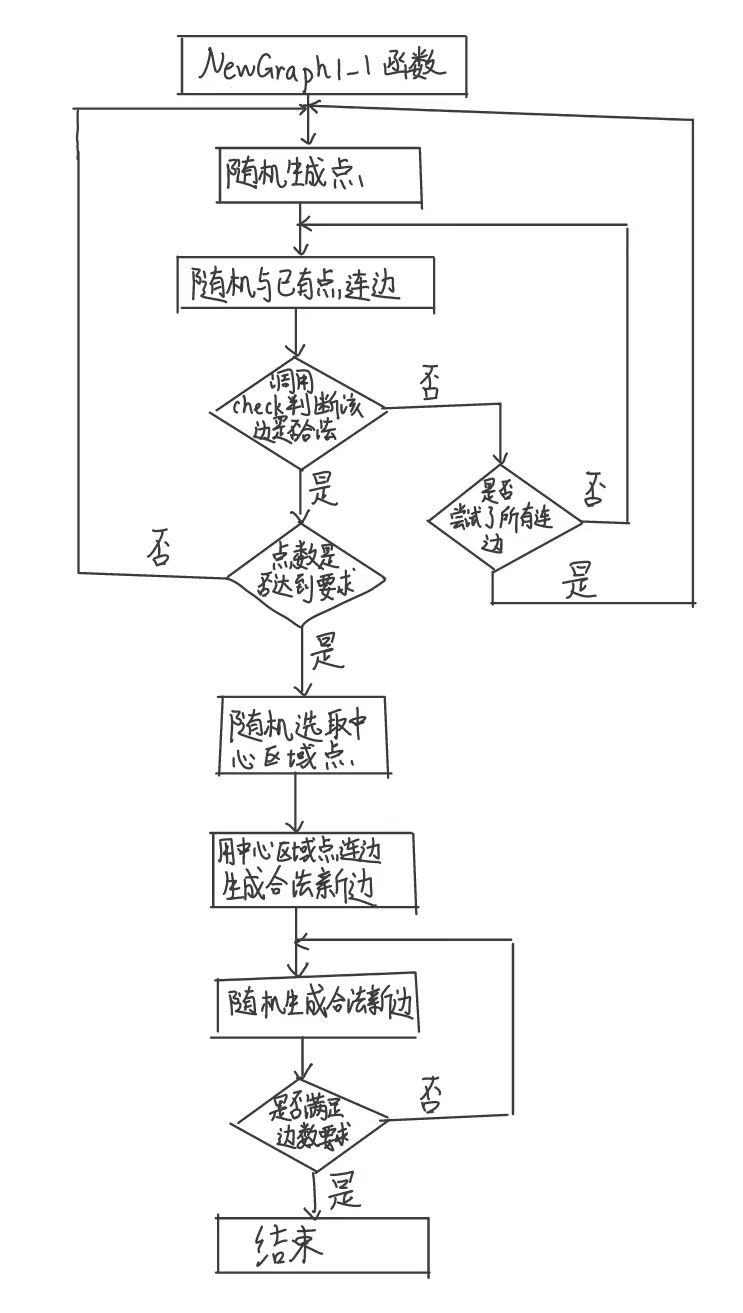
三、过程论述

作为后端代码实现的负责人之一。我主要完成了点，线（边）,图类的实现，编写了NewGraph1\_1，add，check，IsConnected，dfs，FindNearestPoint，Dij，update\_flow，LeastCrossTime，dist,judge,meet等函数的实现。

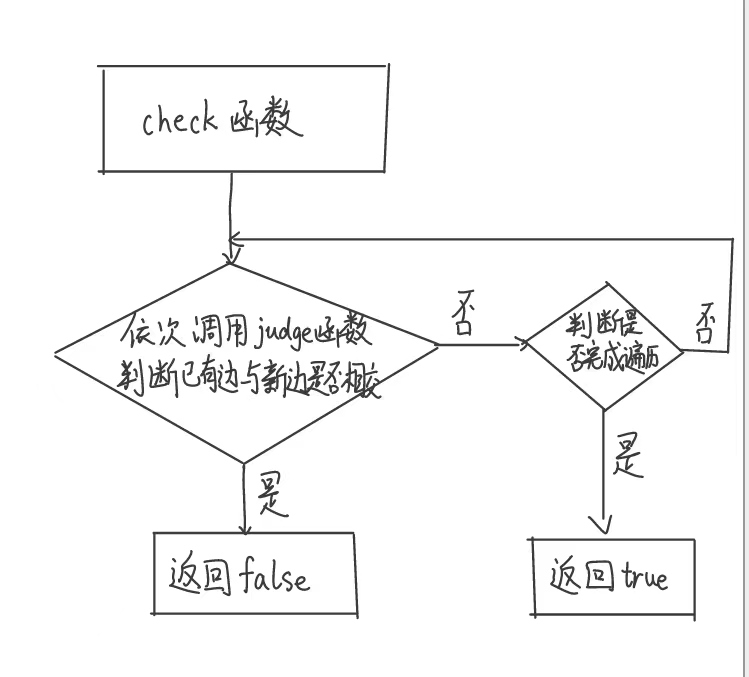
下面我将介绍部分重要函数的具体实现和流程设计。

对于NewGragh1\_1函数，它的实现流程如下图：



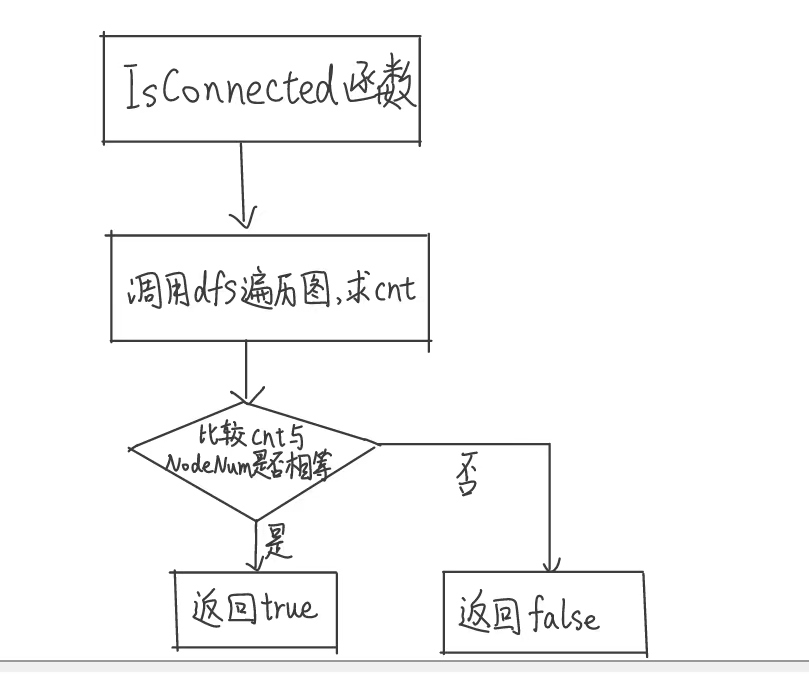
NewGragh1\_1函数的功能是生产新的图。首先随机生成点，并随机生成新边，调用check函数判断新边是否合法，如果不合法，判断是否已经尝试所有新边，如果是，重新生成该点，反之，继续生成新边直达合法或者尝试所有新边。生成合法点和新边后，判断是否符合点数要求，不符合就继续生成点。否则，就进入下一步，选取随机中心区域点，并用中心区域点连边产生新的合法边。最后随机生成合法边直到符合边数要求。

对于check函数，它的实现流程如下图：



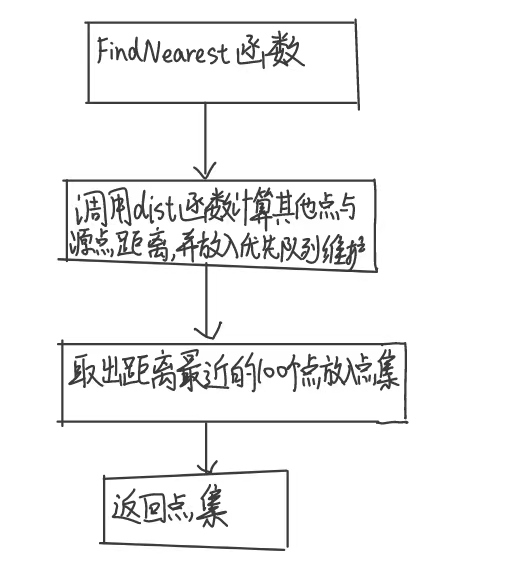
check函数通过调用在点类中实现的croos函数求叉积，利用计算几何知识判断两条线段是否有交点。

对于IsConnected函数，它的实现流程如下图：



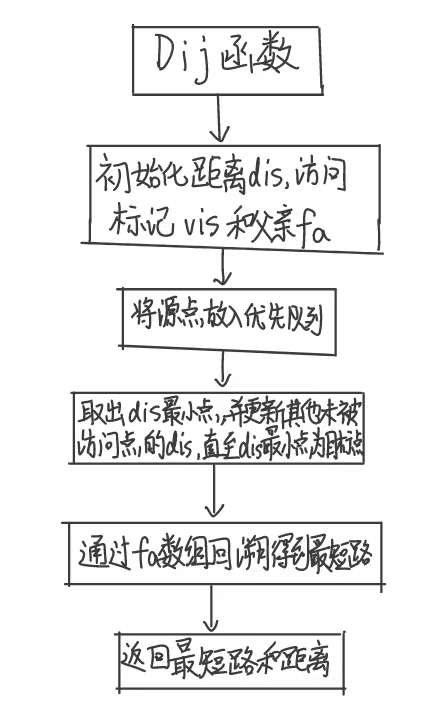
IsConnected函数的功能是判断生成的图是否是连通图。该函数通过调用dfs函数，遍历这幅图得到一次遍历能够访问的点数cnt。将cnt和图中总点数NodeNum比较，相同说明图是连通的，反之，就是不连通的。

对于FindNearest函数，它的实现流程如下图：



FindNearest函数的功能是查找一个点最近的100个点，这个问题可以抽象为距离最小的k的点的问题。采用dist函数快速计算出其他点距离源点的距离，用优先队列来维护最近的100个点，最终返回一个点集。

对于Dij函数，它的实现流程如下图：



Dij函数的功能是计算任意两点之间的最短距离和求最短路径。我采取了dijstra算法，并采取优先队列去维护目前距离源点最近且未被更新的点，优化了时间复杂度到nlogm。采取这个优化是因为该图至少有100000个点，采取暴力的时间复杂度是n^2,时间开销太大，是我无法接受的。为了求最短路径，我维护了一个fa数组，记录点u是由fa[u]更新而来，那么在求得最短路距离后，采取回溯的方法，可以求得最短路径，时间复杂度为O(n).

对于LeastCrossTime函数，它的功能是求任意两点之间的最短通行时间。依旧是采取dijstra算法，通过cross\_time函数计算路径上的通行时间去代替原算法中的距离计算出当前路况下，任意两点之间到达的最短时间和路径。因此实现过程与dig函数相似，详情请参看dij函数流程图。

重点说明设计是如何实现的，包括：对设计工作的详细表述。要求层次分明、表达确切。

要求：每个图都必须有文字说明，图前说明为什么使用该图、图的主要作用；图后说明图中各成分的作用，和成分之间的交互或图所表达的流程。

（正文格式：宋体，小4号，不加粗，两端对齐，行距为固定值20磅）

四、结果分析

对研究过程中所获得的主要的数据、现象进行定性或定量分析，得出结论和推论。（正文格式：宋体，小4号，不加粗，两端对齐，行距为固定值20磅）

五、课程设计总结

总结可以包括:课程设计过程的收获、遇到的问题，遇到问题解决问题过程的思考、程序调试能力的思考，课程设计实现过程中的收获和体会等。