**位置相关服务：附近的出租车**

**项目实习报告**

**姓名：**韩宇

**日期**：2017.7.29

目录

[一、项目概述 3](#_Toc488859455)

[二、系统设计 3](#_Toc488859456)

[2.1. 设计思路 3](#_Toc488859457)

[2.2. 系统框架图 3](#_Toc488859458)

[2.3. 模块功能介绍 3](#_Toc488859459)

[三、功能设计分工 3](#_Toc488859460)

[四、代码设计 3](#_Toc488859461)

[4.1. 数据结构及接口设计 3](#_Toc488859462)

[4.2. 程序流程图 3](#_Toc488859463)

[五、运行结果 4](#_Toc488859464)

[六、心得体会 4](#_Toc488859465)

# 项目概述

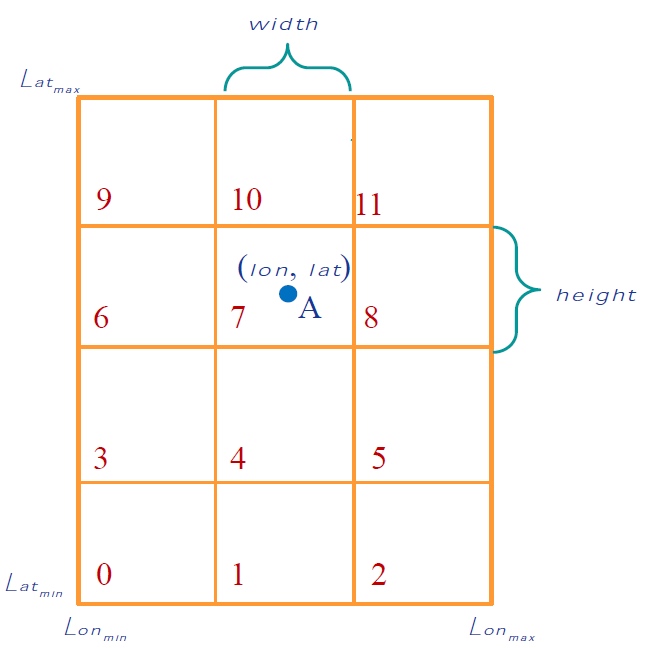
位置相关服务，Location-Based Services，简称 LBS，利用定位技术获得移 动终端（人）的位置信息，并通过通信网络向移动终端（人）提供与位置相关的 信息服务。体系架构包括了定位设备、移动终端、通信网路、移动定位服务中心 和中心数据库。其中的关键技术又囊括了五个方面，分别是定位技术、位置更新 技术、索引技术、查询技术和隐私保护技术。本项目以北京市出租车移动轨迹数 据作为切入点，实现项目“位置相关服务：附近的出租车”，目的是实现一个基 于位置信息的查询服务，通过位置相关的持续查询技术，实时回答用户：“我附 近的出租车”。

# 二、系统设计

## 2.1. 设计思路

系统设计的目的是简单地实现出租车位置的更新与查询，各部分的设计思路 如下：

1.基于网格的索引。索引是一种常用的工具，利用索引可以有效地管理大量 的数据，并快速从数据中查找到特定的数据。根据项目性质，采取划分网格并为 网格建立索引的方式。将所限制的区域依照经纬度中的距离划分为横纵大小相同 的若干个网格，并为其编号，每个网格中的出租车以双向循环链表的形式存入对 应的索引单元，因此，出租车在某区域的进入、离开，都会使双向链表发生增减。



其中，1.空间区域是一个[Ilon, Ilat]大小的空间，Ilon = [Lonmin, Lonmax]，Ilat = [Latmin,Latmax] ；

2.将空间区域划分为row × col个相同大小的网格，每一个小格子称为Cell；

3.网格的列宽width = (Lonmax －Lonmin)/col；

4.网格的行高height = (Latmax －Latmin)/row；

5.Cell Id = col × (lat - Latmin) / height + (lon- Lonmin) / width。

2、需要哈希表（Hash Table）来定位当前出租车所在的网格编号。哈希表的作用是通过确定记录的存放位置和关键字（k）之间的对应关系，方便地 根据记录的关键字检索到对应记录的信息，要构造合适的哈希函数 H（k）使哈 希地址尽可能均匀地分布在连续的内存单元地址上，我们选用除留取余法构建哈 希表，除了纵向延伸哈希表外，横向接入的是含有同关键字车辆信息的双向链表。

3、出租车位置的更新，基于网格索引的查询。模拟真实情况中根据用户定位的位置，给出与其位 置最接近的出租车信息，即实现最近邻查询。此过程先要运用位图索引（Bitmap） 分析搜索过程中是否已遍历到某个网格区域，再利用小顶堆算法（Heap）找出所 有网格中离设定点最近的出租车。根据出租车的位置算出其新的cell\_id，再判断它是新出现的出租车，如果是，就为其分配链表节点p；如果不是，先修改p中的出租车信息，再判断其所在的网格位置是否发生改变。若发生了变化（cell\_id！=cell\_id0），则要将其指针p在原来的cell\_id0的链表中删除，并将其指针添加到新的cell\_id对应的链表中。

为了实现出租车对应cell\_id的快速查找，我们引入哈希表（Hashtable）来实现更新查找功能。哈希表通过哈希函数H(k)建立了出租车地址与其所在cell\_id的对应关系，实现了快速查找。因此，上述步骤可以表述为：

1) 计算出租车位置 (lon ，lat) 对应的 cell\_id；

2) 根据出租车唯一标识号id在哈希表中查找该出租车的上一次更新cell\_id0以及链表指针p0, 如果没有找到，转 5）；

3) 直接修改p0所指向出租车状态信息后，如果 cell\_id0等于cell\_id，返回；否则，继续；

4) 更新 Hash 节点的 cell\_id，将 p0 从cell\_id0的双向循环俩表中删除，并将 p0 添加到 cell\_id 的双向循环链表中；返回；

5) 重新分配链表节点 P0, 赋值之后将 P0 插入到 cell\_id的双向循环链表中，更新 Hash 表。

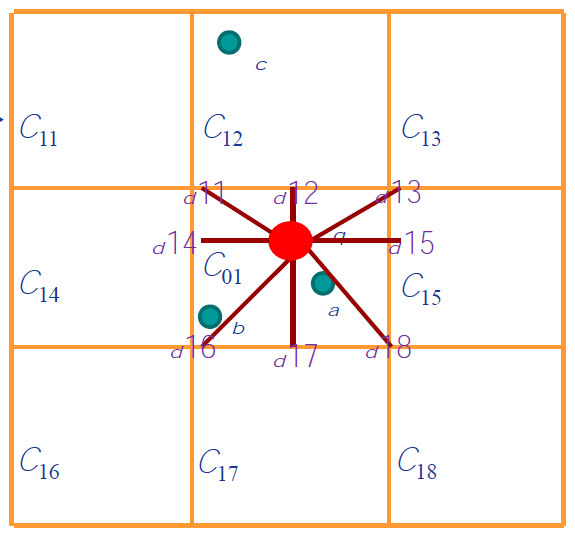
3、查询某一范围内所有出租车信息

找到用户附近的出租车，就要实现查询某一给定范围内所有出租车信息的功能。首先要通过给定的经纬度范围算出所包含的cell\_id，再遍历每个cell\_id中所有出租车，即实现了该功能。

4、最近邻查询

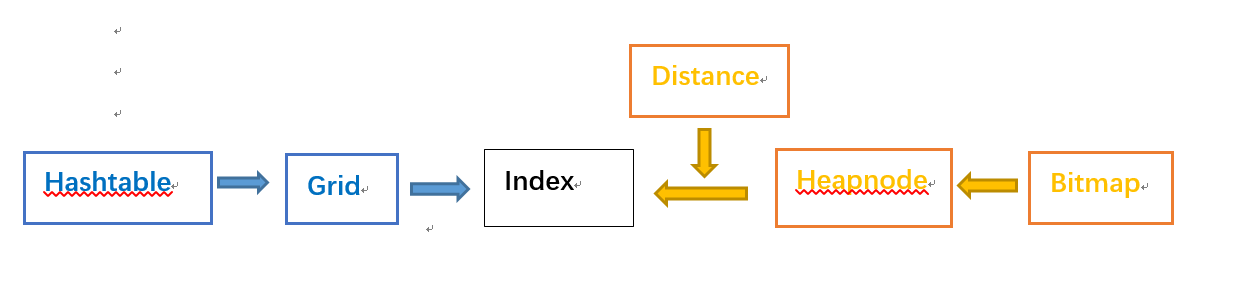
最后要实现的功能就是查询距离用户最近的出租车，即找所有出租车中到用户距离最小的那一个。这里采用的方法是小顶堆排序法找到距离最近的出租车。具体步骤如下：

1. 计算查询点q所在的网格C01并计算q与C01的距离，依据 q与C01的距离创建小顶堆；
2. 将堆顶元素(C01)弹出，若为空间点，则该点就是所找的最近点，计算结束；若为网格，则找出网格对应的空间点，依据q与空间点的距离，将空间点插入小顶堆中，并调整小顶堆，再进行步骤3）；
3. 计算以被删网格 (C01)为中心的周围8个网格（C11 , C12 ,…, C18 ），并计算 q与周围网格（C11 ,C12 ,…, C18 ）的距离（分别为 d11 , d12 ,…, d18）。依据距离，将这八个网格（C11 , C12 ,…, C18）插入小顶堆中，并调整小顶堆；
4. 重复进行步骤2），直至计算结束。



从上述过程中可以看出，并不是每次都需要加入八个格子的。

## 2.2. 系统框架图



## 2.3. 模块功能介绍

1. 索引更新模块。具体操作如下：

1）进行网格（grid）与哈希表（Hashtable）的构建和初始化；

2）计算出租出位置，即经纬度（lon, lat）的计算，以及改坐标对应落入的网格编号（cell\_id）;

3)根据id 在哈希表中查找该出租车上一次更新的网格编号cell\_id0以及链表指针p0，如果没有找到，则重新分配链表节点p0，赋值之后将p0插入到cell\_id的双向链表中，更新哈希表；

4）直接修改链表指针所指向的出租车状态信息后，如果cell\_id0等于cell\_id，则返回值；若不是，则继续;

5)更新哈希表节点的网格编号，将p0从上一个网格编号的双向链表中删除，并将p0添加到cell\_id的双向循环链表中，返回；

2.索引查询模块。具体操作如下：

1）计算查询点 q 所在的网格 C01，计算 q 与 C01 的最小距离，为 0。依据 q 与 C01 的距离，创建小顶堆；

2）将堆顶元素弹出(C01)，若为网格则找出网格对应的空间点并删除该网格， 依据 q 与空间点的距离，将空间点插入小顶堆中，并调整小顶堆；

3）计算以被删网格(C01)为中心的周围 8 个网格（C11, C12, …, C18），计算 q 与网格 C11, C12, …, C18 的最小距离，分别为 d11, d12, …, d18。依据最小距 离，将 C11, C12, …, C18 插入小顶堆中，并调整小顶堆；

4）计算以被删网格(C12)为中心的周围 8 个网格（新增 C’11, C’12, C’13 三 个区域），计算 q 与新增网格 C’11, C’12, C’13 的最小距离，依据最小距离，将新 增的网格插入小顶堆中，并调整小顶堆；

5）将堆顶元素弹出(a)，若为空间点则找出的是最近邻的空间点，查找结束。 否则，继续重复上述步骤。

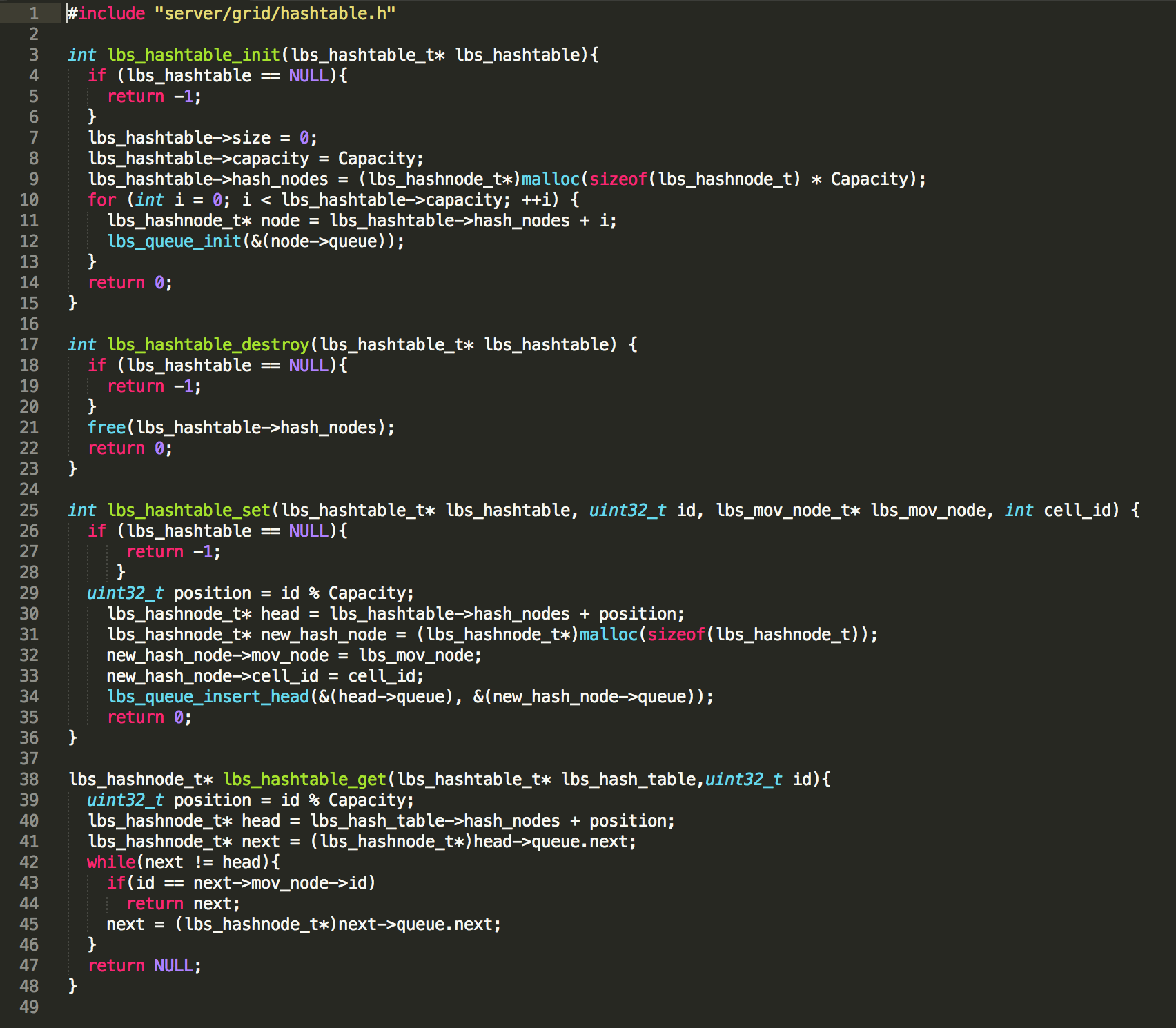
# 三、功能设计分工

项目小组为 3 人小组，组长陈佳琳，组员金镕、韩宇。其中陈佳琳负责网格模块（Grid）的编写，金镕负责小顶堆模块（Heap）的编写，我负责哈希表（Hash Table）、位图索引模块（Bitmap）距离（distance）模块的编写。此外，三人共同完成模块衔接（Index）的编写。

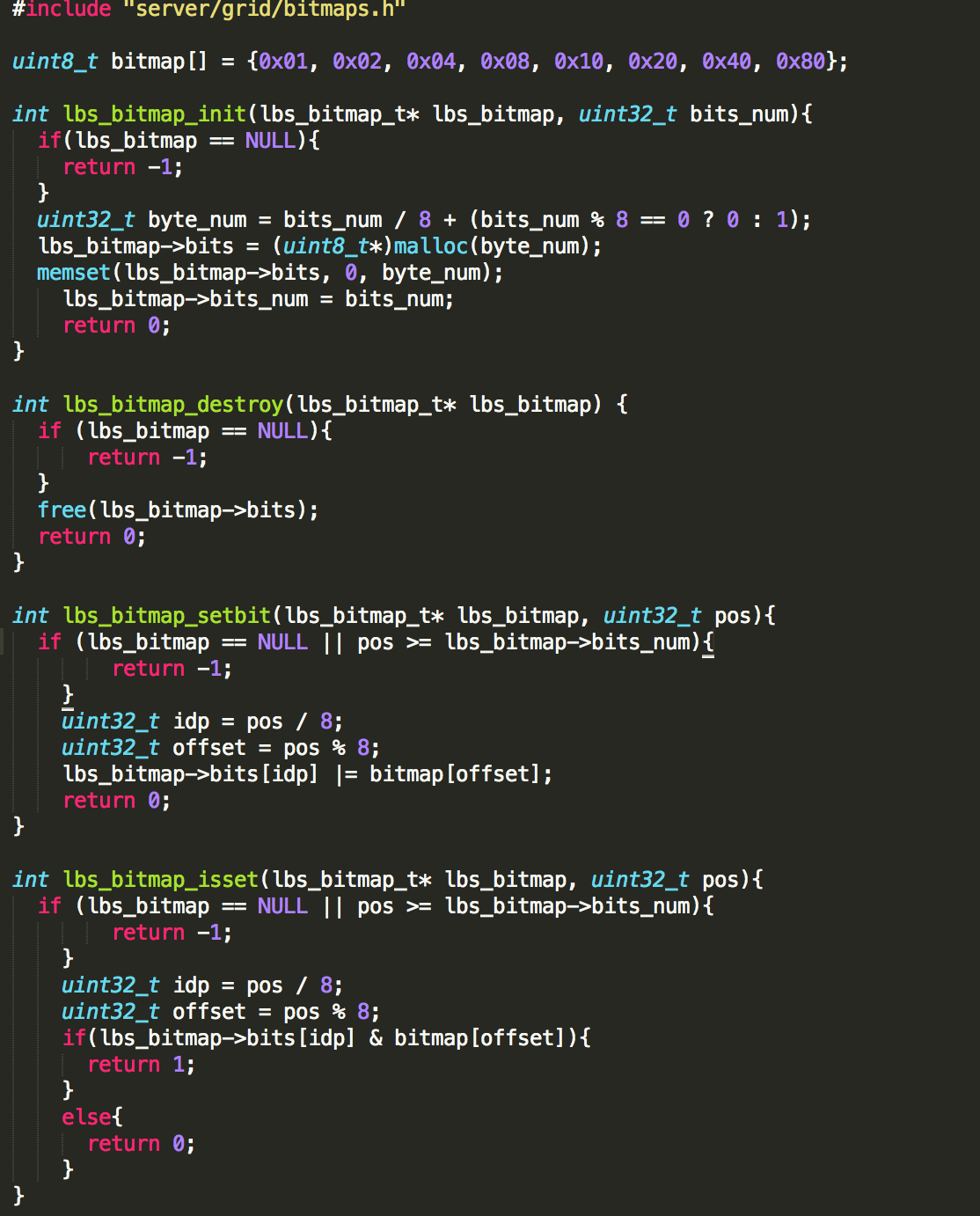
# 四、代码设计

## 4.1. 数据结构及接口设计

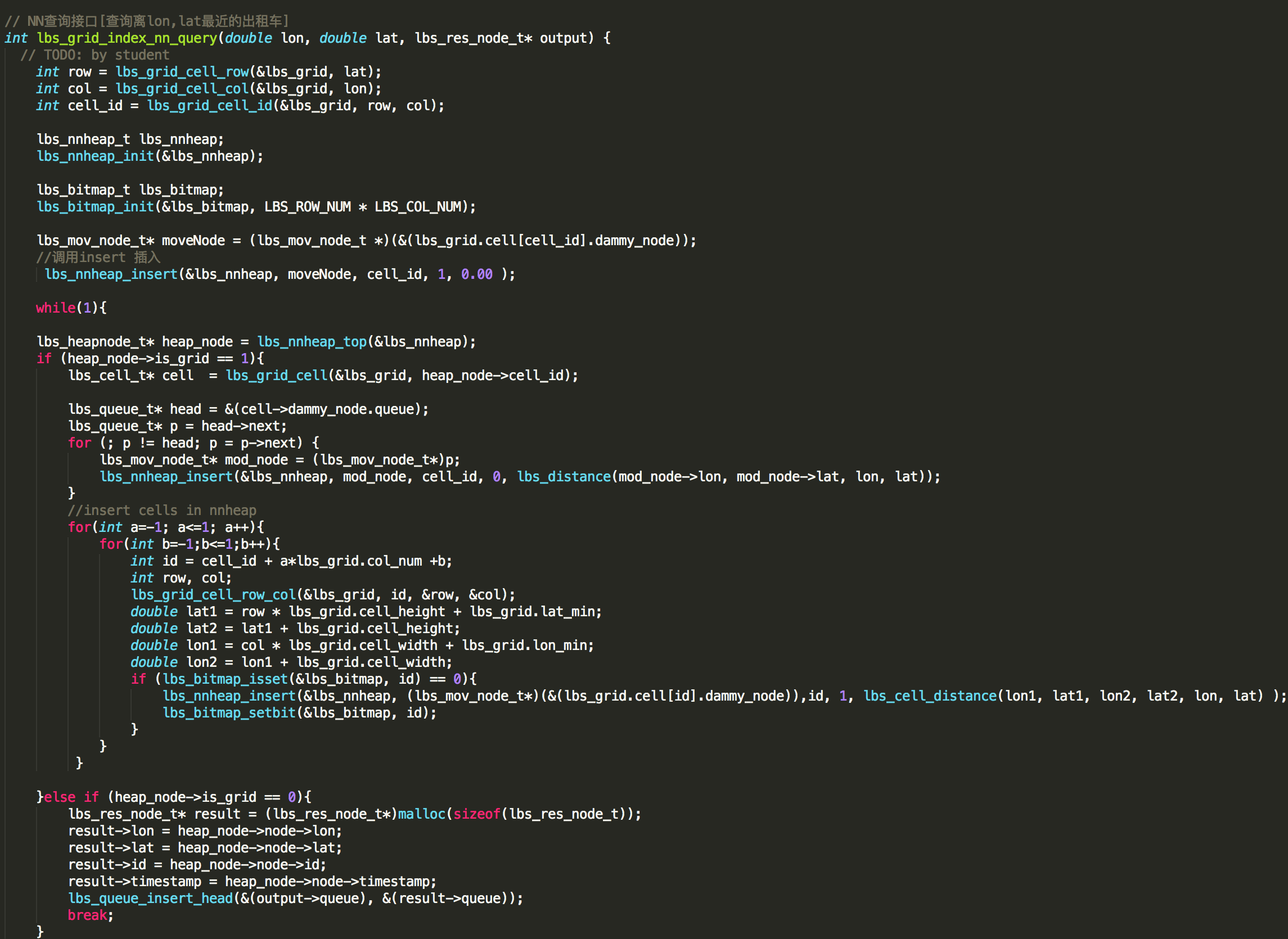
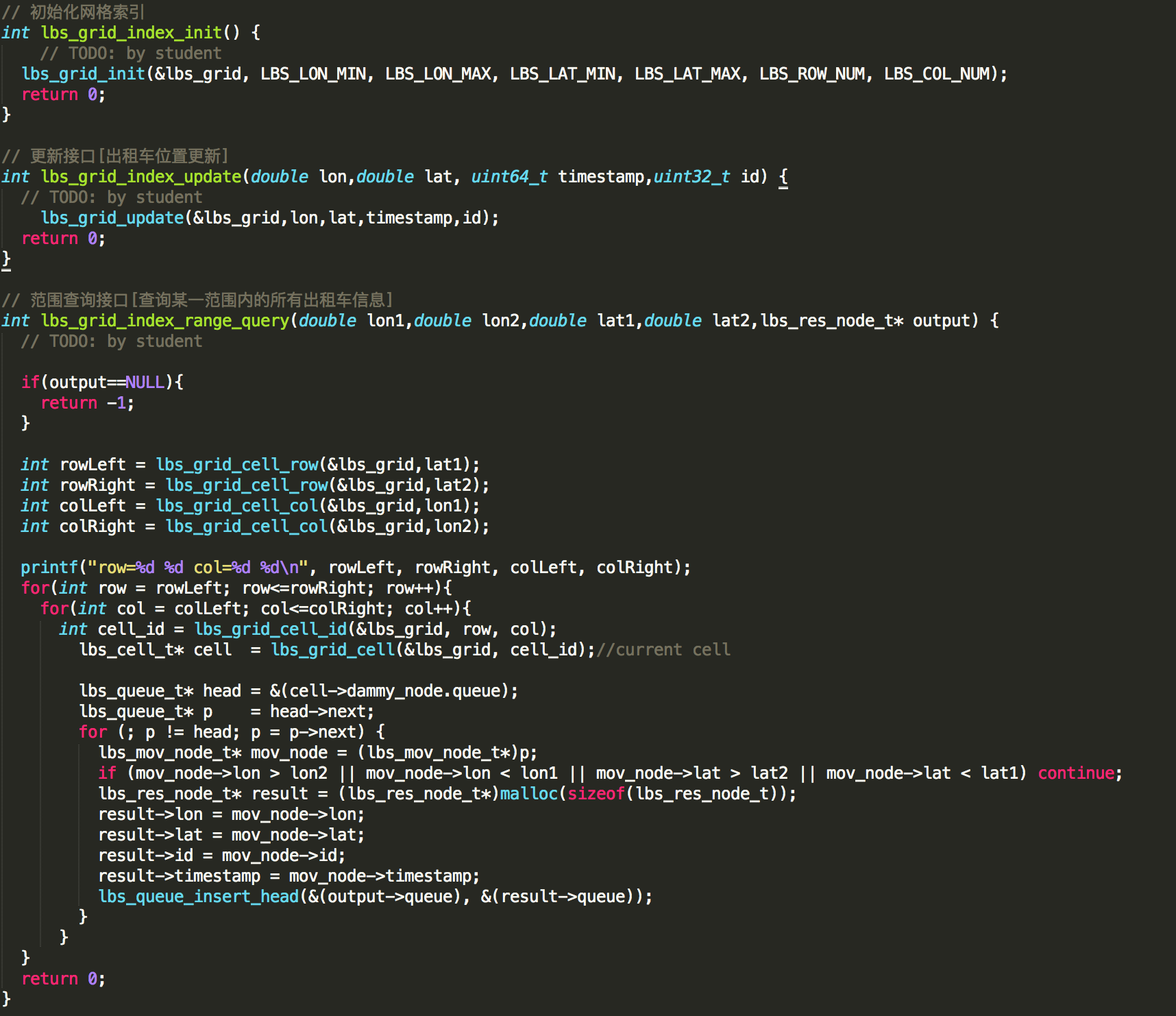
Hashtable



Bitmap:

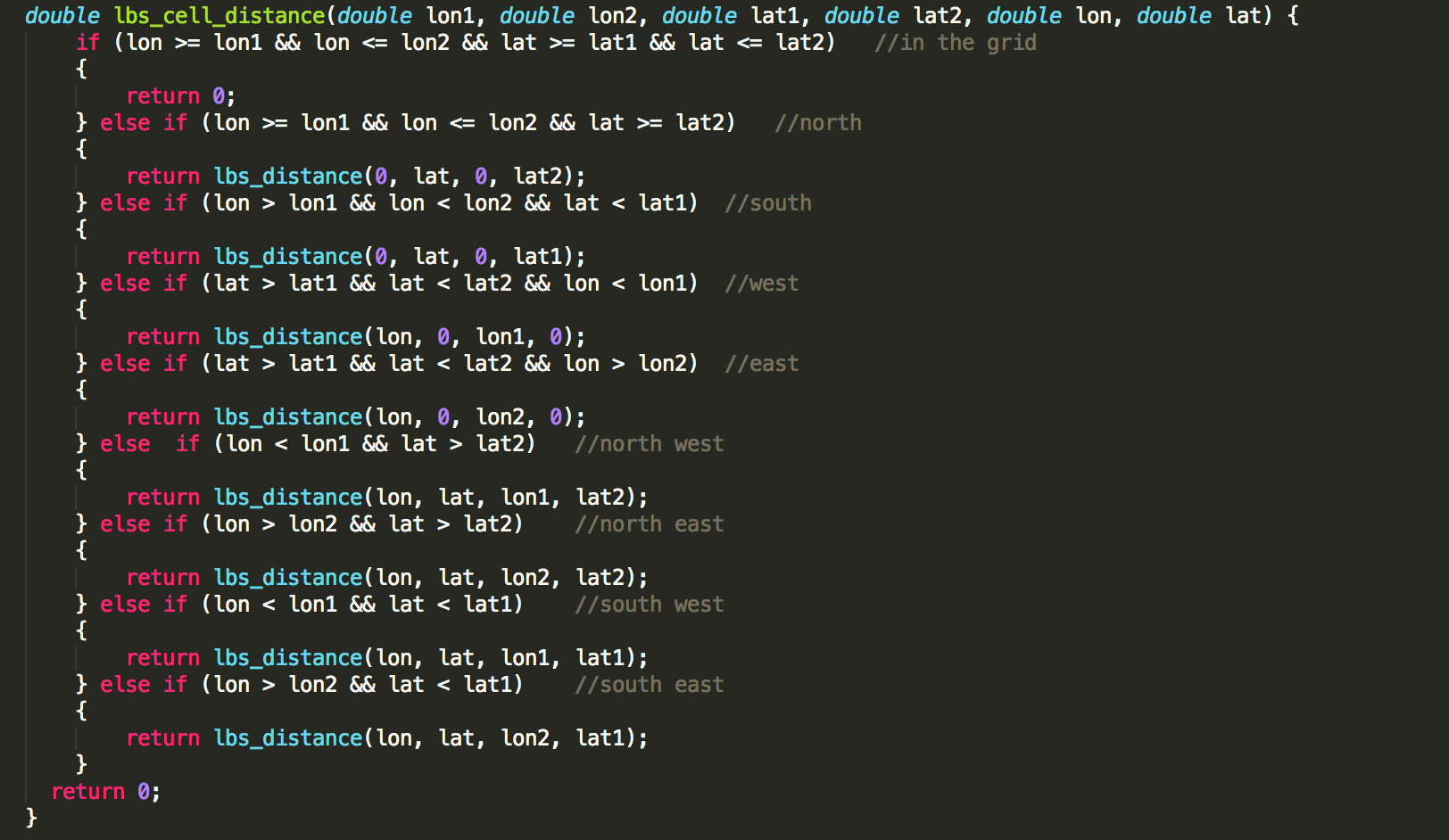


Index:



## 4.2. 程序流程图

Distance:



按照上北下南左西右东的方式进行判定

开始

是

点在网格右边？

是

否

否

点在网格上边？

点在网格上边？

点到网格的距离等于点到网格右上角点的距离

是

点到网格的距离等于点到网格右上角点的距离

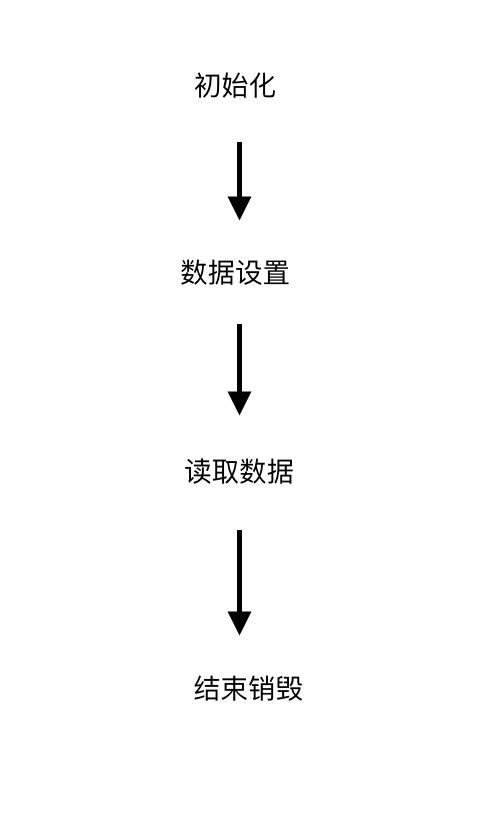
是

点到网格的距离等于点到网格右上角点的距离

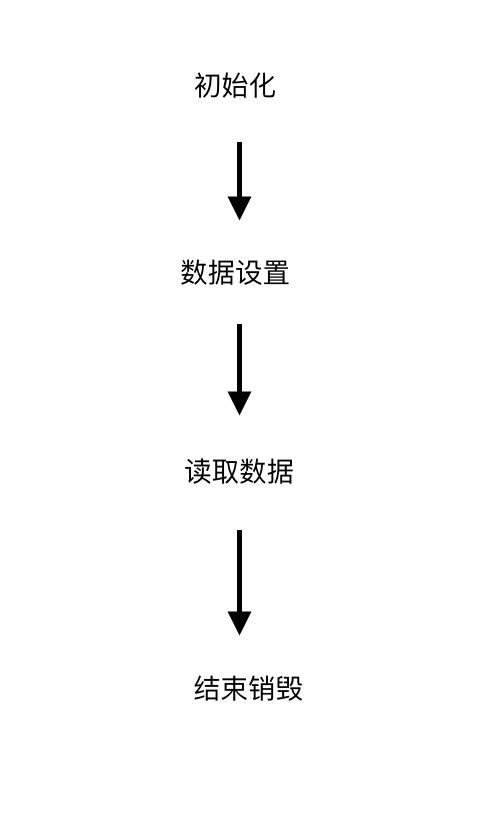
点到网格的距离等于点到网格右上角点的距离

结束

Hash table：



Bitmap：



# 五、运行结果



# 六、心得体会

通过参加这次位置相关服务项目实习，我收获很多。在程序设计方面更加灵活，与组员分工合作，效率变高的同时思考方式也变得多样化，不会像以前一样的自己埋头苦干。程序的设计更加有结构，而不是将所有的源码全部塞到一个main文件中。小组合作也变得更加协调，没有之前的死板，个人只完成个人的任务。