**华中科技大学网络空间安全学院**

**程序设计综合课程设计**

**报告**

题目: 移动终端与GSM蜂窝网络的通信模拟

班 级： 信安21XX班

学 号： U2021XXXXX

姓 名： MTX

成 绩：

指导教师： 吴俊军

**完成日期：** 2023**年** 3 **月** 13 **日**

**目 录**

[一、系统需求分析 1](#_Toc452449527)

[二、总体设计 7](#_Toc452449528)

[三、数据结构设计 9](#_Toc452449529)

[四、详细设计 14](#_Toc452449530)

[五、系统实现 31](#_Toc452449531)

[六、运行测试与结果分析 38](#_Toc452449532)

[七、总结 45](#_Toc452449533)

[八、参考文献 48](#_Toc452449534)

# 一、系统需求分析

## （一）.主要功能

### 1.数据的存储，首个和最后分块基站数据的显示

（1）功能阐释：构建一个分块/四叉树来存储基站信息，方便高效查找指定位置附近的基站数据。

（2）输入数据：无

（3）输出数据：将基站信息进行分块储存，并输出首个分块（最左上角）的基站信息，最后分块（最右下角）的基站信息

（4）系统应满足的功能要求和性能要求：首先我选择的是分块的处理方式，为提高程序的可移植性，以便将来可能遇到更多种类基站，我认为分块的大小应该是动态可调节的，于是设计时在头文件宏定义中定义了每个分块的大小，考虑到此题高速基站的有效半径在5km左右，而城区和乡镇的有效半径基本小于等于1km，我采用两种分块方式，第一种将所有基站按1km\*1km分块，第二种单独将高速基站按5km\*5km分块，由此减少了后续查找信号最强的基站这一功能的时间复杂度。定义每个分块中平均有n个基站，故时间复杂度为O(n)。

### 2.显示特定分块的基站数据

（1）功能阐释：显示出西北角分块/树叶的东侧分块/树叶基站数据以及南侧相邻分块/树叶的基站数据,显示出东南角所在分块/树叶的西北侧基站数据，以及西北侧的北侧相邻分块/树叶的基站数据。

（2）输入数据：无

（3）输出数据：四块指定分块的基站数据

（4）系统应满足的功能要求和性能要求：通过合理的大小进行分块后，能够依次输出西北角分块、以及南侧相邻分块、东南角所在分块、以及西北侧的北侧相邻分块的基站信息。定义每个分块中平均有n个基站，故时间复杂度为O(n)。

### 3.找到指定坐标附近最强信号的基站信息

（1）功能阐释：给定坐标(X，Y)，找到距离该点信号强度最强的基站，并输出该基站的信息。

（2）输入数据：特定的坐标(X，Y)

（3）输出数据：距离(X，Y)信号强度最强的基站的信息

（4）系统应满足的功能要求和性能要求：根据输入的坐标(X，Y)，找到其对应的分块，并遍历其周围的8个分块，在这9个分块中找到信号最强的基站，并输出该基站的信息。定义每个分块中平均有n个基站，故时间复杂度为O(n)。

### 4.判断指定坐标是否被基站覆盖

（1）功能阐释：给定坐标(X，Y)，当所在为主均不在附近基站的有效信号范围内，则说明该位置没有被基站信号覆盖。

（2）输入数据：特定的坐标(X，Y)

（3）输出数据：若有基站覆盖此坐标，则输出距离(X，Y)信号强度最强的基站的信息；若无基站覆盖此坐标，则输出无基站覆盖

（4）系统应满足的功能要求和性能要求：根据输入的坐标(X，Y)，找到其对应的分块，并遍历其周围的8个分块，在这9个分块中找到信号最强的基站，并输出该基站的信息；若九个分块中没有基站覆盖，则输出无基站覆盖。定义每个分块中平均有n个基站，故时间复杂度为O(n)。

### 5.终端移动时基站的切换情况

（1）功能阐释：给定移动终端的移动轨迹，计算需要不断切换到哪些基站进行通信。

（2）输入数据：移动轨迹查询的步长

（3）输出数据：每段移动轨迹对应的基站切换情况

（4）系统应满足的功能要求和性能要求：用户输入一个步长，对于任意一段轨迹，我们将终点到起点的连续路径按照步长进行密集的均分，用密集的离散的点代替一段连续的轨迹，再查找每个点对应的信号强度最强的基站，当相邻的点对应的信号强度最强的基站不同时，代表进行了一次切换，从而我们可以输出整段轨迹对应的基站切换情况。该算法的性能和输入的步长有关，当步长越小，越容易保证轨迹的基站切换情况，但算法复杂度也就越高，当步长过大时，有可能会漏掉基站切换信息。经过多次实验，我发现步长输入为10米时较为合适。

## （二）.扩展功能

### 1.判断第1段移动轨迹进入/离开首个基站的时间信息

（1）功能阐释：检查第1段移动轨迹，分析进入首个基站信号有效范围的时间，分析离开首个基站信号有效范围的时间。

（2）输入数据：要查询的轨迹编号

（3）输出数据：该段轨迹进入的首个基站的编号，以及进入和离开时间

（4）系统应满足的功能要求和性能要求：任务仅要求输出第1段轨迹进入和离开首个基站信号有效范围的时间，但我将此功能一般化，根据用户输入想要查询的移动轨迹，程序都能输出该段轨迹进入的首个基站的编号，以及进入和离开时间。根据主要功能5，我们可以得知每段轨迹进入的第一个基站的编号，于是我们只需判断何时进入和离开此我们已知的基站，故时间复杂度为O(1)。

### 2.判段第3、6段轨迹处于首个信号重叠区的的时间长度

（1）功能阐释：分析第3段、第6段开始阶段切换基站信号重叠区，计算通过重叠区的时间长度。

（2）输入数据：要查询的轨迹编号

（3）输出数据：信号重叠区的两个基站的编号，以及该轨迹在信号重叠区的时间

（4）系统应满足的功能要求和性能要求：任务仅要求输出第3段轨迹和第6段轨迹处于首个信号重叠区的的时间长度，但我将此功能一般化，根据用户输入想要查询的移动轨迹，程序都能输出段该轨迹处于首个信号重叠区的的时间长度。根据主要功能5，我们可以得知每段轨迹进入的第一个基站的编号和第二个基站的编号，于是我们只需判断何时进入和离开此我们已知的前两个基站，故时间复杂度为O(1)。

## （三）.升级功能

### 1.第12段轨迹在速度较低时连接上伪基站的情况

（1）功能阐释：检查第12段移动轨迹，在移动终端移动速度较低的情况下连接上近距离的伪基站。

（2）输入数据：要查询的轨迹编号

（3）输出数据：从具体X时刻开始连接Y编号的伪基站T秒

（4）系统应满足的功能要求和性能要求：任务仅要求判断第12段轨迹是否连接上为基站，若连接上则输出，连接上的时刻、伪基站的编号、以及连接的时间长度，但我将功能扩展为所有轨迹都可以查询该功能。用户输入查询的轨迹后，我们将运动轨迹终点到起点的连续路径进行密集的均分，用密集的离散的点代替一段连续的轨迹，同时将伪基站的运动轨迹也用密集的点代替，以时间为变量，遍历每一个时刻，查找是否有伪基站和移动终端距离第一次在40m以内，若有则记录时间节点T1，记录该伪基站的编号N，并记录下一次距离大于40m的时间节点T2，则X=T1，Y=N，T=T2-T1，输出这三个量即可。

### 2.第9段轨迹在速度较高时连接上伪基站的情况

（1）功能阐释：检查第9段移动轨迹，移动终端移动速度较快，但与伪基站保持较短距离，短时间连接上伪基站。

（2）输入数据：要查询的轨迹编号

（3）输出数据：从具体X时刻开始连接Y编号的伪基站T秒

（4）系统应满足的功能要求和性能要求：任务仅要求判断第9段轨迹是否连接上为基站，若连接上则输出，连接上的时刻、伪基站的编号、以及连接的时间长度，但我将功能扩展为所有轨迹都可以查询该功能。用户输入查询的轨迹后，我们将运动轨迹终点到起点的连续路径进行密集的均分，用密集的离散的点代替一段连续的轨迹，同时将伪基站的运动轨迹也用密集的点代替，以时间为变量，遍历每一个时刻，查找是否有伪基站和移动终端距离第一次在40m以内，若有则记录时间节点T1，记录该伪基站的编号N，并记录下一次距离大于40m的时间节点T2，则X=T1，Y=N，T=T2-T1，输出这三个量即可。

# 二、总体设计

移动终端与GSM蜂窝网络的通信模拟系统总体分为4个模块：信息的存储和预处理模块、基站的选择模块、移动轨迹处理模块、伪基站处理模块。（如图2-1所示）

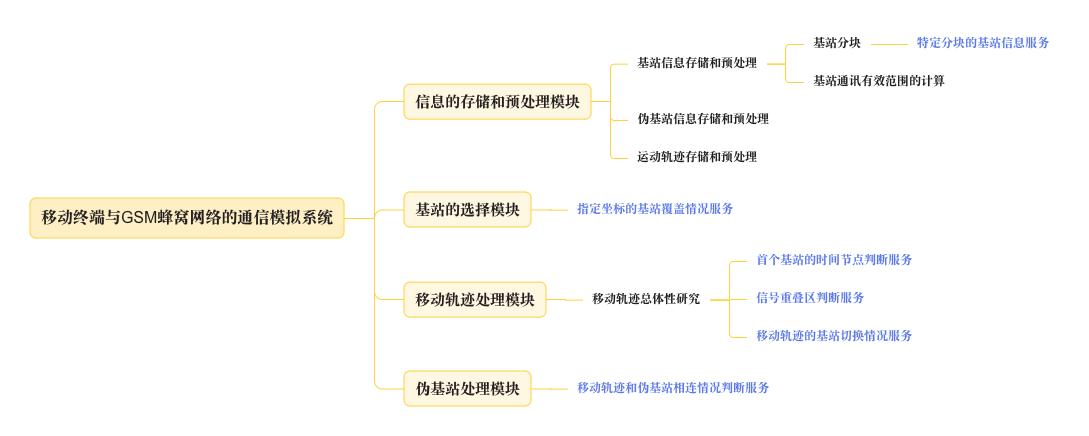


图2-1 移动终端与GSM蜂窝网络的通信模拟系统框架图

## 1.信息的存储和预处理模块

信息的存储：从文件中读取基站、移动轨迹、伪基站的相关信息并用合适的数据结构进行存储。

预处理：对于基站信息，我们将基站进行分并我们将根据基站的类型和基站的强度计算出基站的信号有效范围并进行存储；对于伪基站数据，我们将其运动的起始时间和终点时间转化为秒为单位并进行存储，获取每个伪基站运动的长度并进行存储。

实现的具体任务：能输出特定分块的基站信息。（对应主要功能1、2）

### 2.基站的选择模块

基站的选择：根据输入的坐标(X，Y)，找到其对应的分块，并遍历其周围的8个分块，在这9个分块中找到信号最强的基站，并输出该基站的信息；若九个分块中没有基站覆盖，则输出无基站覆盖。从而完成了对应坐标的基站选择功能。（对应主要功能3、4）

### 3.移动轨迹处理模块

移动轨迹的基站切换情况：用户输入一个步长，对于任意一段轨迹，我们将终点到起点的连续路径按照步长进行密集的均分，用密集的离散的点代替一段连续的轨迹，再查找每个点对应的信号强度最强的基站，当相邻的点对应的信号强度最强的基站不同时，代表进行了一次切换，从而我们可以输出整段轨迹对应的基站切换情况。（对应主要功能5）

首个基站的时间节点判断：根据用户输入想要查询的移动轨迹，程序输出该段轨迹进入的首个基站的编号，以及进入和离开时间。（对应扩展功能1）

信号重叠区判断：根据用户输入想要查询的移动轨迹，程序都能输出该段轨迹处于首个信号重叠区的的时间长度，以及重叠区的两个基站的编号。（对应扩展功能2）

### 4.伪基站处理模块

移动轨迹和伪基站相连情况判断：检查第12段移动轨迹，在移动终端移动速度较低的情况下连接上近距离的伪基站；检查第9段移动轨迹，移动终端移动速度较快，但与伪基站保持较短距离，短时间连接上伪基站，输出从具体X时刻开始连接Y编号的伪基站T秒。（对应升级功能1、2）

# 三、数据结构设计

1.基站的数据结构

采用一维数组：static Jz jz[MAX\_JZ\_NUMBER]

具体Jz结构体中的成员（如图3-1所示）

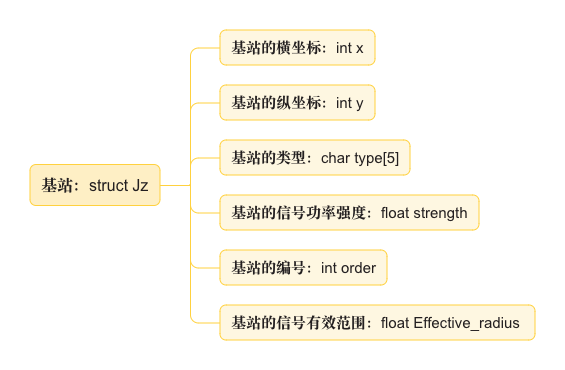


图3-1 基站的结构体说明图

图3-1解释说明：基站的横坐标、纵坐标、类型、信号功率强度、编号为文件中给出的信息，而基站的有效范围是我为方便后续功能的实现自行添加的，其计算公式为：Effective\_radius=a\* sqrtf(strength)，当基站类型为“城区”、“乡镇”、“高速”时，a依次取300、500、1000。

数据机构的用法：一维数组直接根据索引便可轻易增删改查，不多赘述。

2.移动轨迹的数据结构

采用一维数组：static Yd yd[MAX\_YD\_NUMBER]

具体Yd结构体中的成员（如图3-2所示）

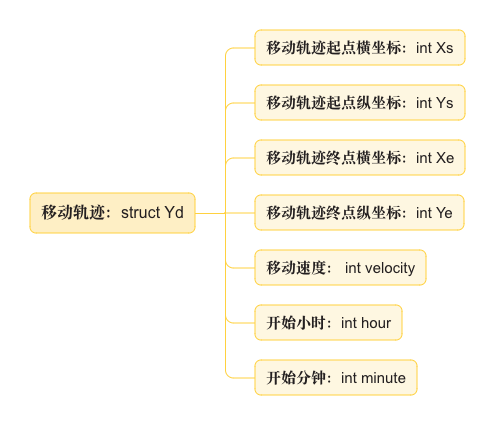


图3-2 终端移动轨迹的结构体说明图

图3-2解释说明：移动轨迹的起点横坐标、起点纵坐标、终点横坐标、终点纵坐标、移动速度、开始小时、开始分钟均为文件中给出的信息，直接从文件中读取即可。

数据机构的用法：一维数组直接根据索引便可轻易增删改查，不多赘述。

3.伪基站移动轨迹的数据结构

采用一维数组：static Wz wz[MAX\_WZ\_NUMBER]

具体Wz结构体中的成员（如图3-3所示）

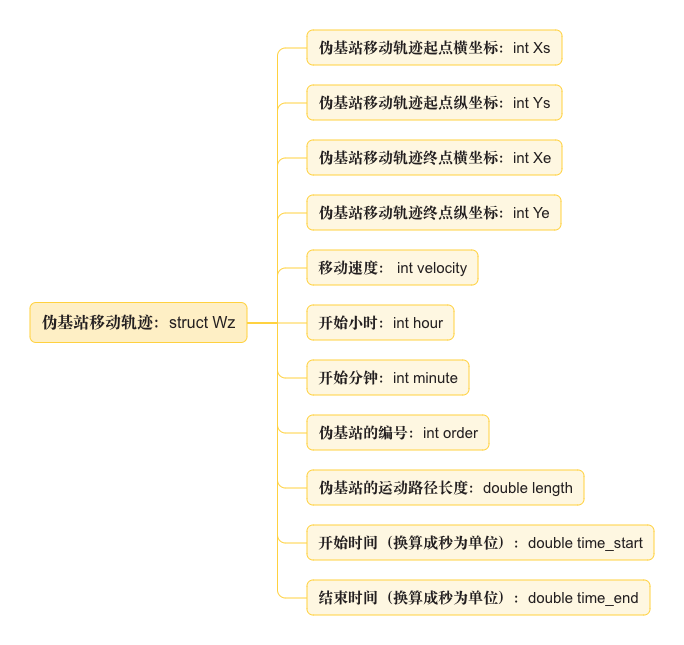


图3-3 伪基站移动轨迹的结构体说明图

图3-3解释说明：伪基站移动轨迹起点横坐标、起点纵坐标、终点横坐标、终点纵坐标、移动速度、开始小时、开始分钟、伪基站的编号为文件中给出的信息，伪基站的运动路径长度、开始时间、结束时间是我为方便后续功能的实现自行添加的。

4.分块的数据结构

考虑到高速基站的有效范围较大，我们将高速单独分块，我们采用两个二维数组：

static District All[MAX\_DISTRICT\_LENGTH\_X\_ALL][MAX\_DISTRICT\_LENGTH\_Y\_ALL]

static District Gs[MAX\_DISTRICT\_LENGTH\_X\_GS][MAX\_DISTRICT\_LENGTH\_Y\_GS]

具体District结构体中的成员（如图3-4所示）

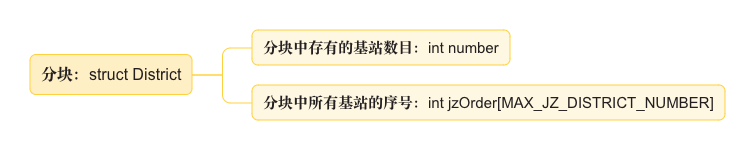


图3-4 分块的结构体说明图

图3-4解释说明：我们以All[MAX\_DISTRICT\_LENGTH\_X\_ALL][MAX\_DISTRICT\_LENGTH\_Y\_ALL]为例来解释是如何通过分块来访问每块中的基站（如图3-5所示）

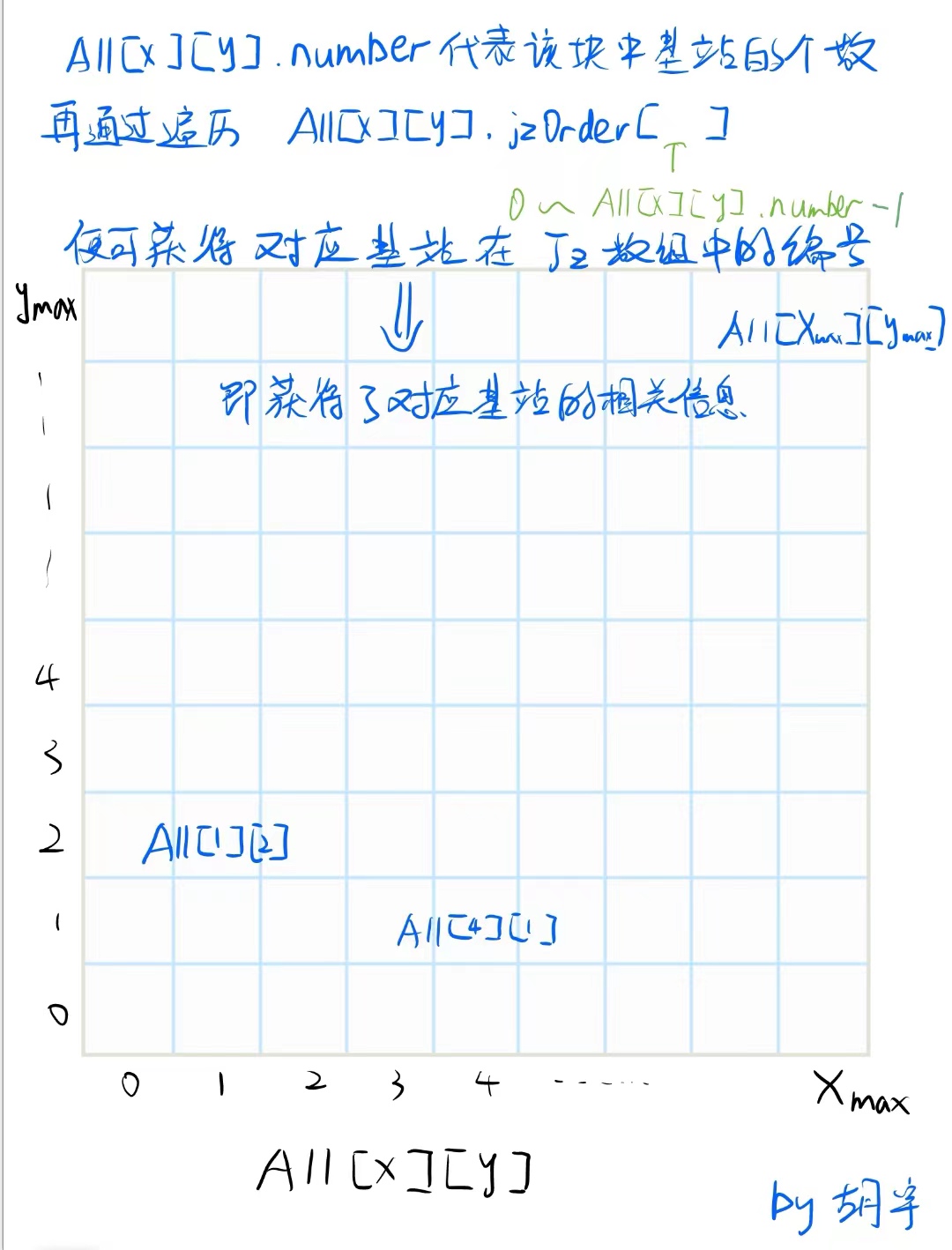


图3-5 分块的具体用法说明图

# 四、详细设计

## （一）.辅助函数算法设计

首先再介绍主体功能实现算法之前，我将先介绍一些基本辅助函数的算法设计。

1.基站信息的存储（如图4-1-1所示）

此函数的作用是将基站信息读入。

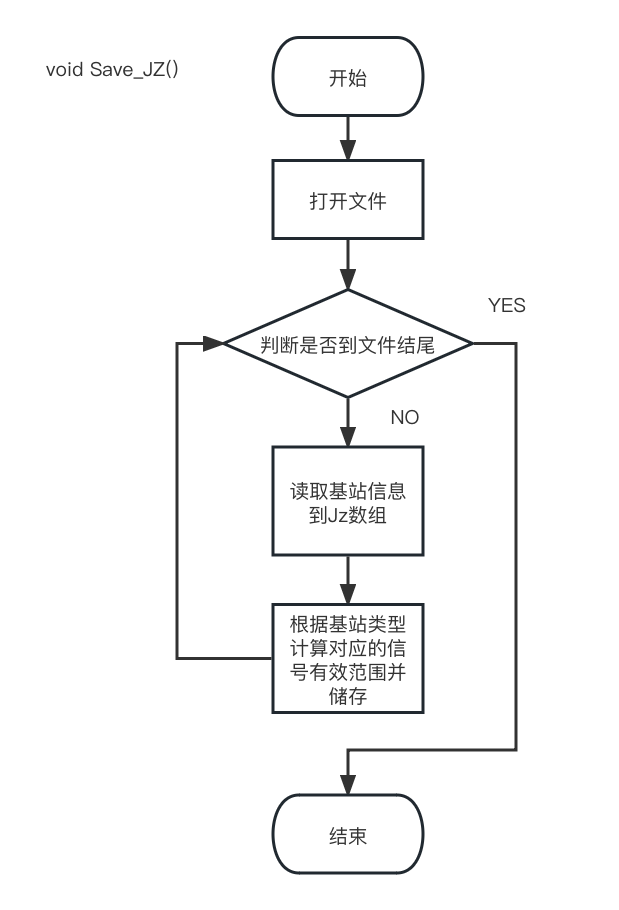


图4-1-1 Save\_JZ函数流程图

2.运动轨迹信息的存储（如图4-1-2所示）

此函数的作用是将终端移动轨迹信息读入。

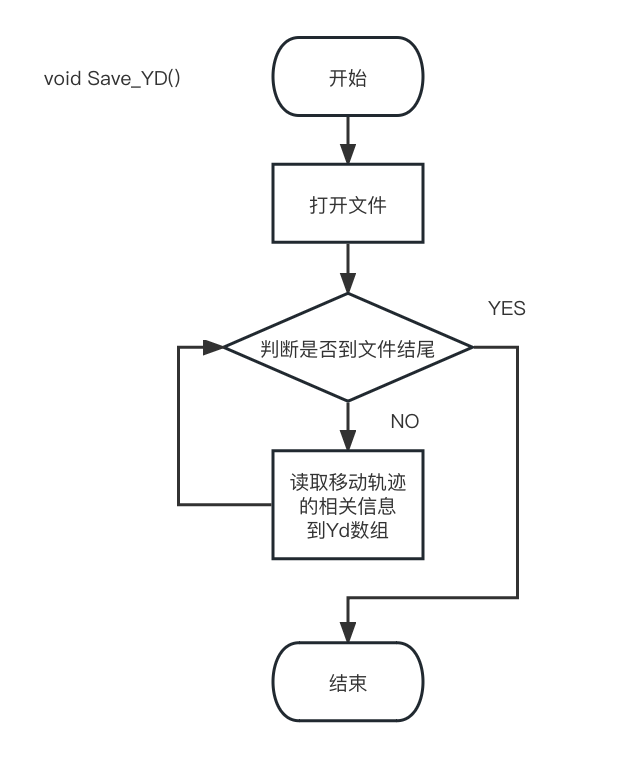


图4-1-2 Save\_YD函数流程图

3.伪基站运动信息的存储（如图4-1-3所示）

此函数的作用是将终端移动轨迹信息读入。

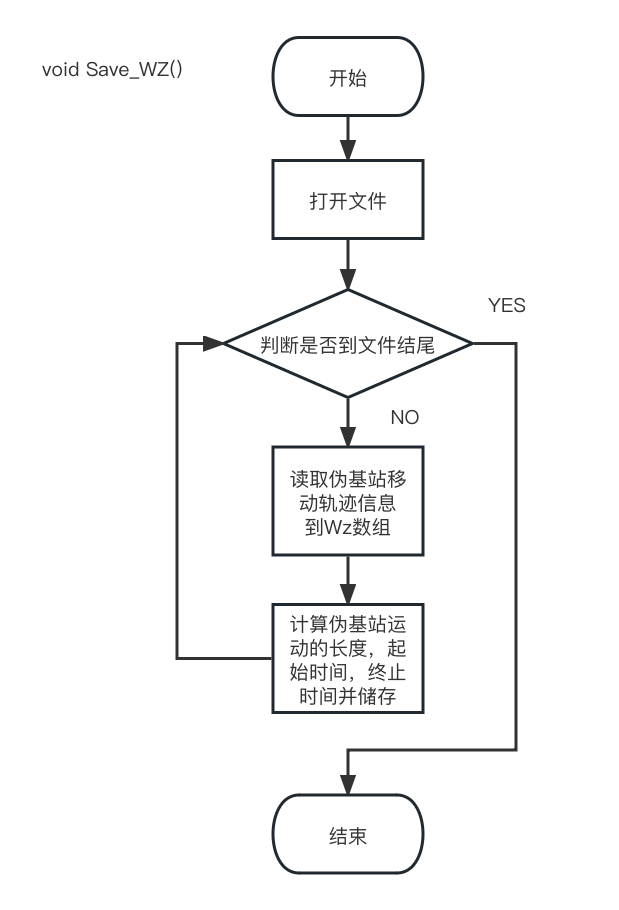


图4-1-3 Save\_WZ函数流程图

4. 获取基站信息坐标的最值（如图4-1-4所示）

此函数的作用是获取所有基站坐标的最值。

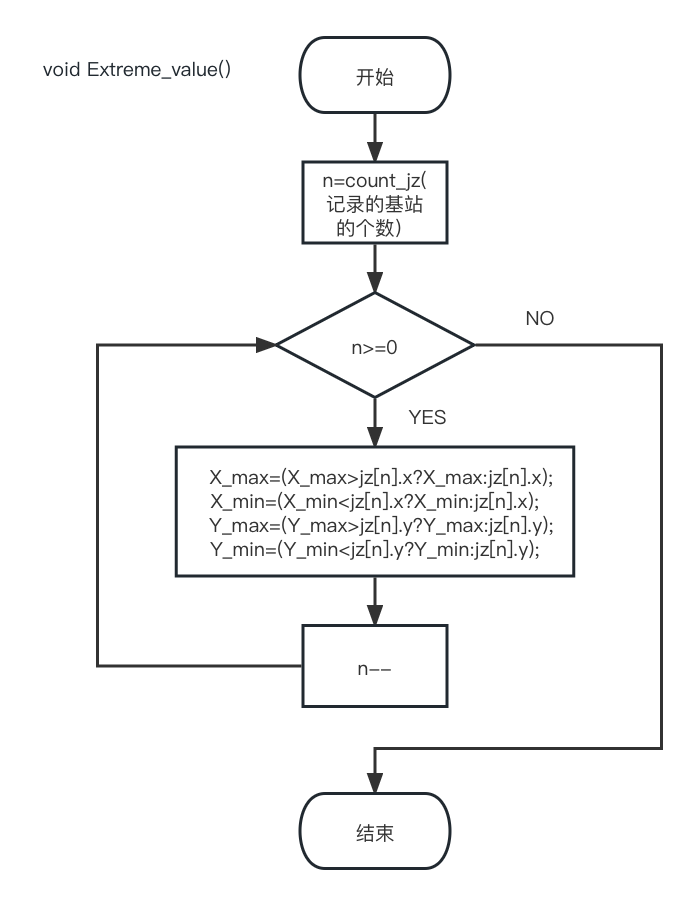


图4-1-4 Extreme\_value函数流程图

5.将基站按方式一分块（如图4-1-5所示）

此函数遍历所有基站信息，根据基站的坐标计算出其对应的分块并存储进该分块，这种分块的大小为1000\*1000。

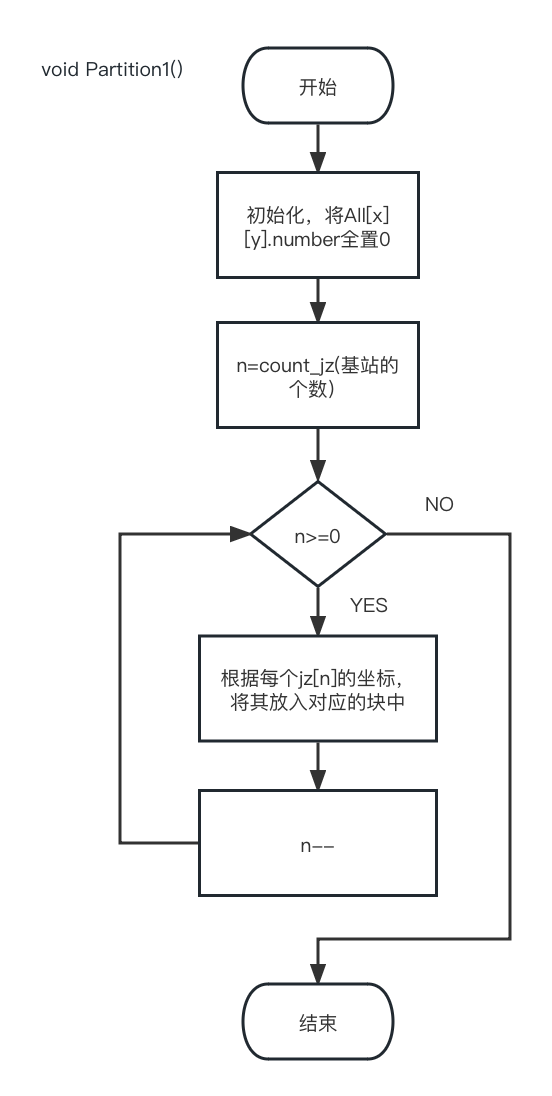


图4-1-5 Partition1函数流程图

6.将基站按方式二分块（如图4-1-6所示）

我经过对三种基站数据的观察，我发现高速的信号有效范围远大于城区和乡镇，于是我按5000\*5000单独将高速分块，而按1000\*1000将所有基站分块，两种分块方式的好处是，在查找信号最强的基站时，遍历的基站数会大大减少，而且能确保结果的可靠性，减少了算法的时间复杂度。

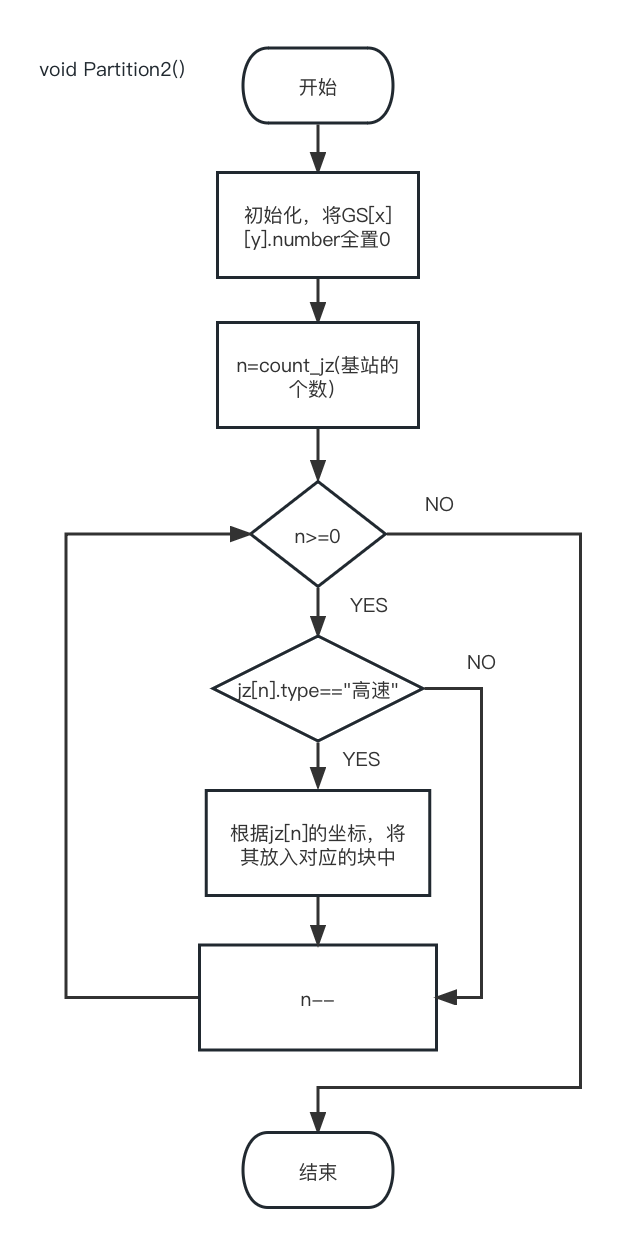


图4-1-6 Partition2函数流程图

7.获取一个分块中的信号最强基站（如图4-1-7所示）

此函数作用是查找在（X，Y）分块中对于坐标（x，y）信号最强的基站，我们遍历在（X，Y）分块中的所有基站，记录其对于（x，y）的信号强度power1，并将power1和之前记录的强度power进行比较，若之前的power较大，则继续遍历，若power1较大，则将power更新为power1，然后继续遍历，直到（X，Y）分块中所有的基站都被遍历完，我们就找到了最强信号的基站，

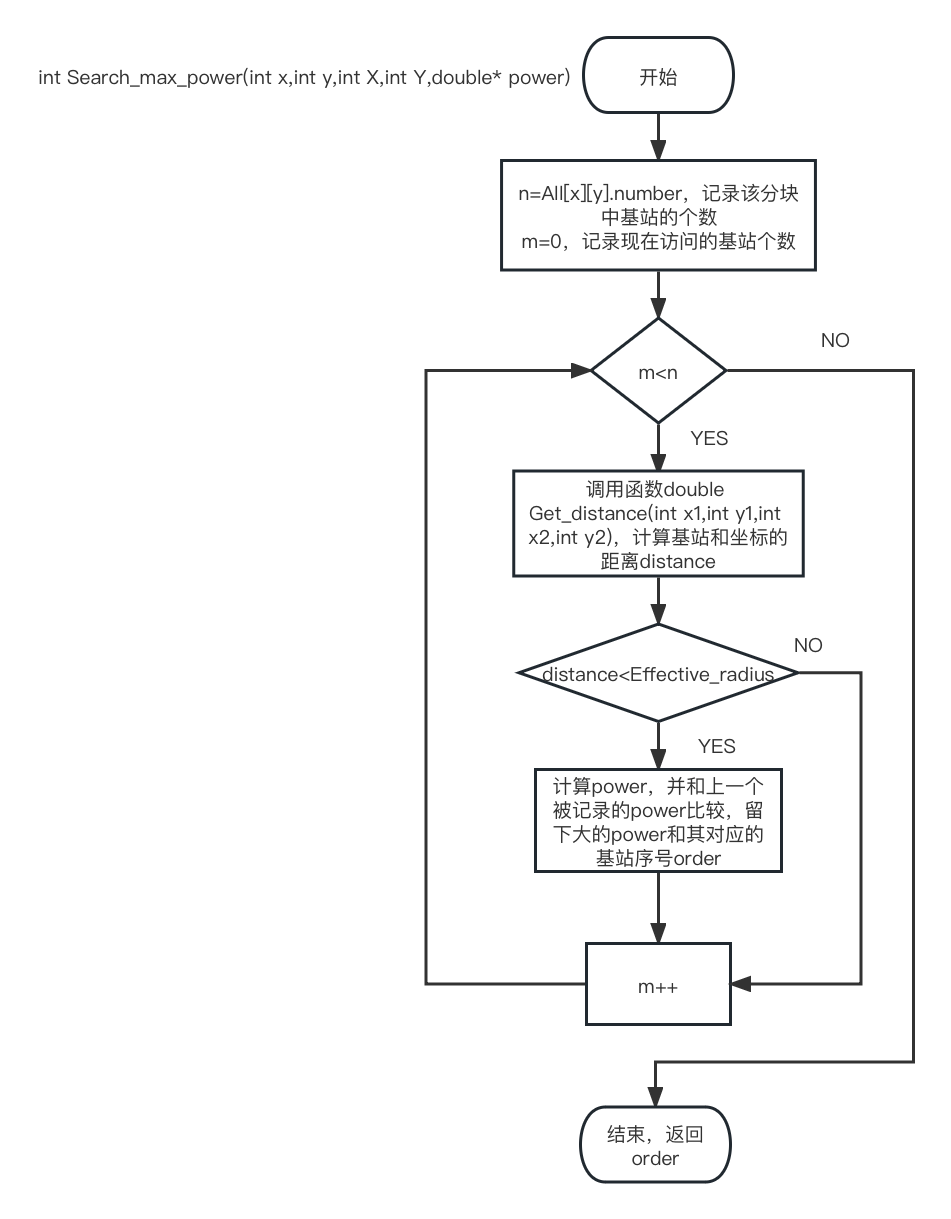


图4-1-7 Search\_max\_power函数流程图

8.获取周围九个分块中信号最强基站（如图4-1-8所示）

此函数作用是获取距离坐标（x，y）信号最强的基站的信息，很容易便想到我们找到（x，y）所在周围九个分块中信号最强的基站即可，我们可借助Search\_max\_power函数获得每个分块中信号最强的基站，再将这9个基站进行比较，我们便可找到距离坐标（x，y）信号最强的基站的信息。

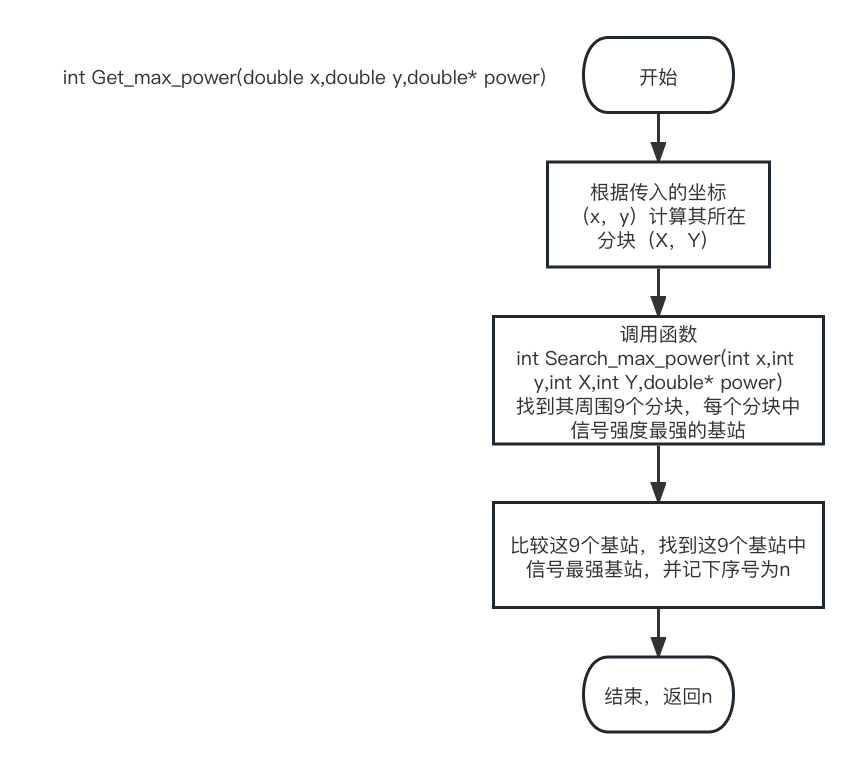


图4-1-8 Get\_max\_power函数流程图

9.获取两坐标间的距离（如图4-1-9所示）

此函数的作用是得到两点间的距离。

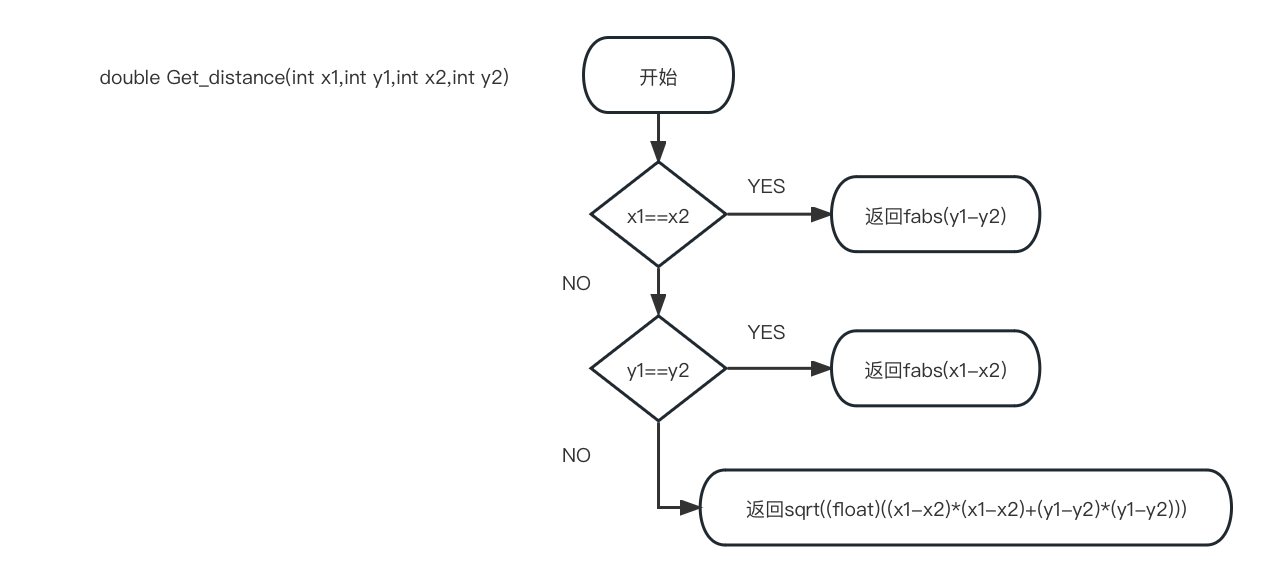


图4-1-9 Get\_distance函数流程图

10.获取移动轨迹的移动方向（如图4-1-10所示）

此函数的作用是得到终端移动轨迹的运动方向。



图4-1-10 Get\_direction\_yd函数流程图

11.获取伪基站的移动方向（如图4-1-11所示）

此函数的作用是得到伪基站移动轨迹的运动方向。



图4-1-11 Get\_direction\_wz函数流程图

## （二）.主体功能算法设计

1.数据的存储，首个和最后分块基站数据的显示（如图4-2-1所示）

该函数作用为依次输出首个分块和最后一个分块的基站信息。

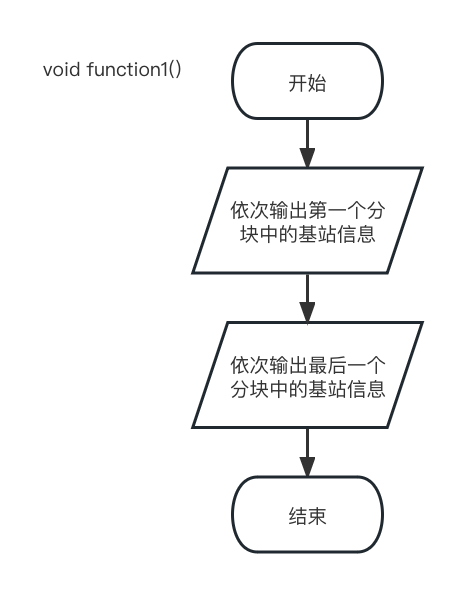


图4-2-1 function1函数流程图

2.显示特定分块的基站数据（如图4-2-2所示）

该函数作用为依次输出四个特定分块的基站信息。

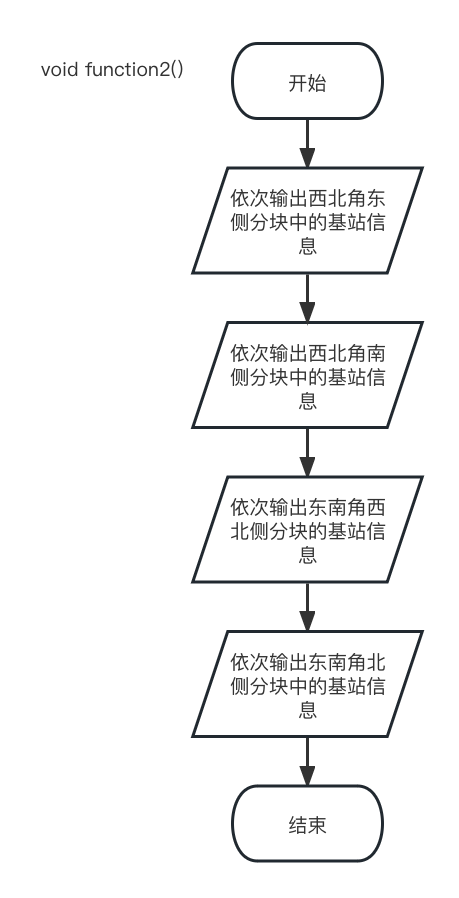


图4-2-2 function2函数流程图

3&4. 判定坐标是否被覆盖，找到指定坐标附近最强信号的基站信息（如图4-2-3所示）

调用Get\_max\_power函数，若返回值是-1，则代表该坐标没有基站覆盖，若不为-1，则返回的是信号最强基站的序号。

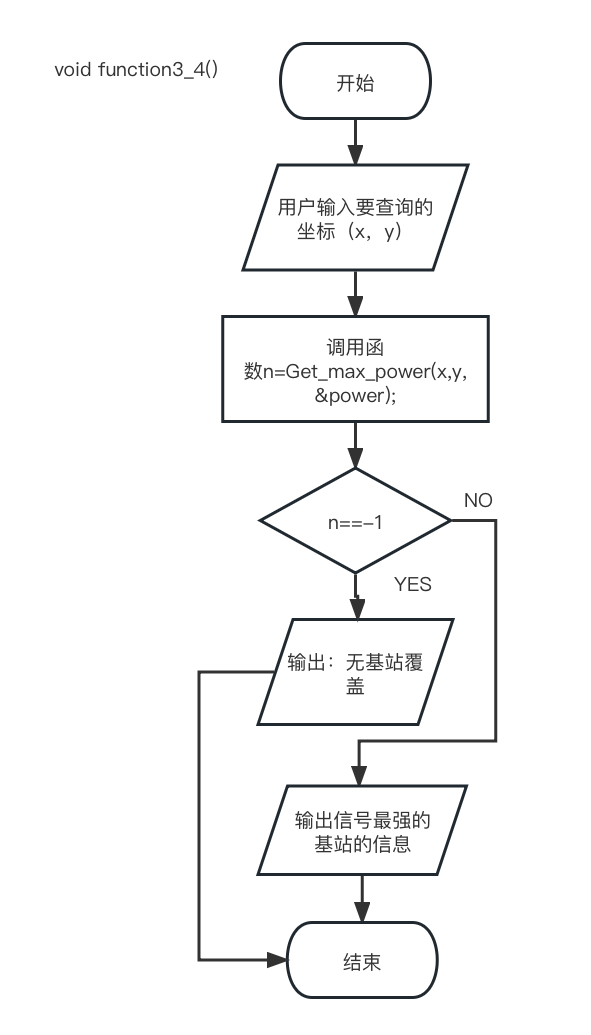


图4-2-3 function3\_4函数流程图

5.终端移动时基站的切换情况（如图4-2-4所示）

此函数，首先根据用户输入设置步长，然后遍历每一条移动轨迹，以该步长逐步移动，记录每刻的信号最强的基站，若相邻两个时刻对应的基站不相同，则代表一次切换，便输出这次切换，直到终端运动到这段轨迹的终点。由此，便可以得到每段轨迹所有的基站切换情况。

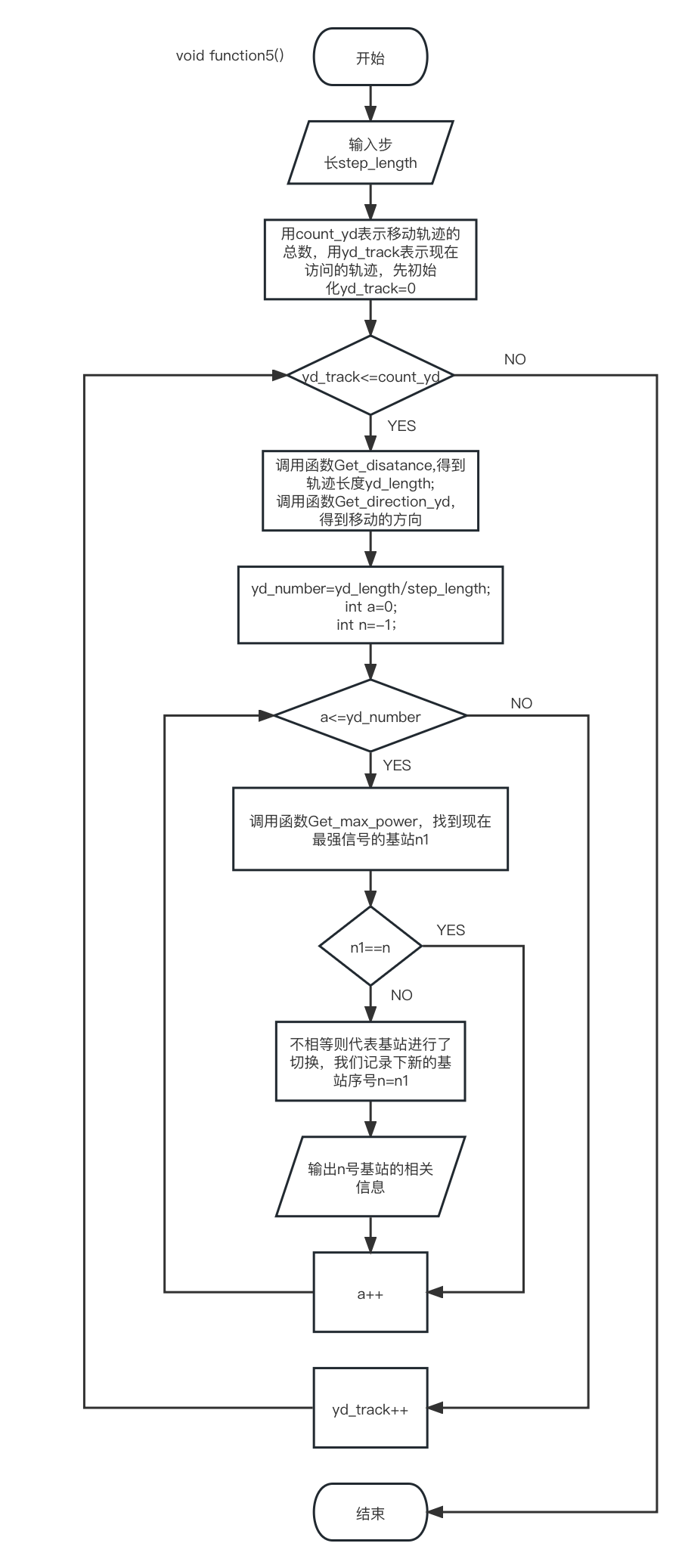


图4-2-4 function5函数流程图

6.判断第1段移动轨迹进入/离开首个基站的时间信息（如图4-2-5所示）

此函数中先判定起点时，终端是否在某基站的信号内，若是，则a1=0，只需记录出这个基站的时间节点a2即可；若不是，则随着时间的增加，记录第一次进入某基站信号范围内的时间节点a1，再随之终端的移动，记录终端第一次出这个基站的时间节点a2，a1、a2便是进入/离开首个基站的时间信息。

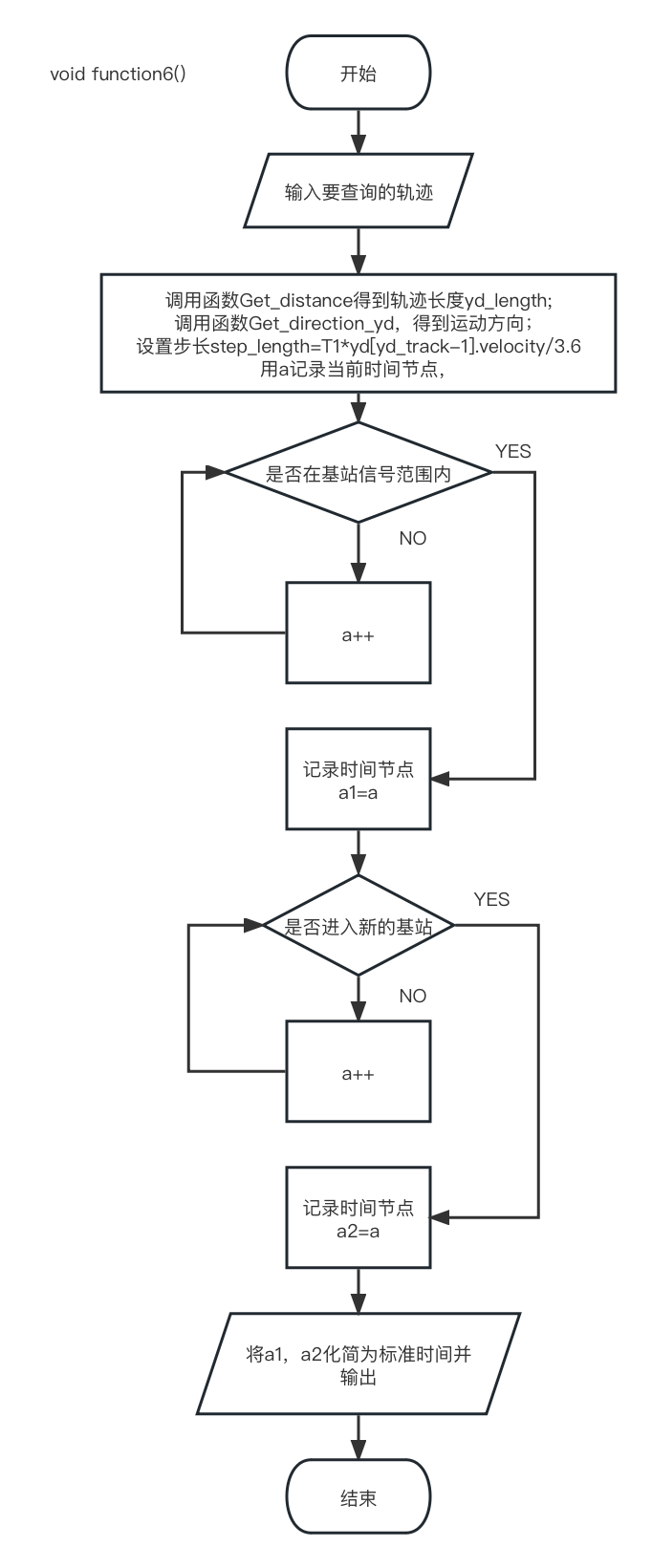


图4-2-5 function6函数流程图

7. 判段第3、6段轨迹处于首个信号重叠区的的时间长度（如图4-2-6所示

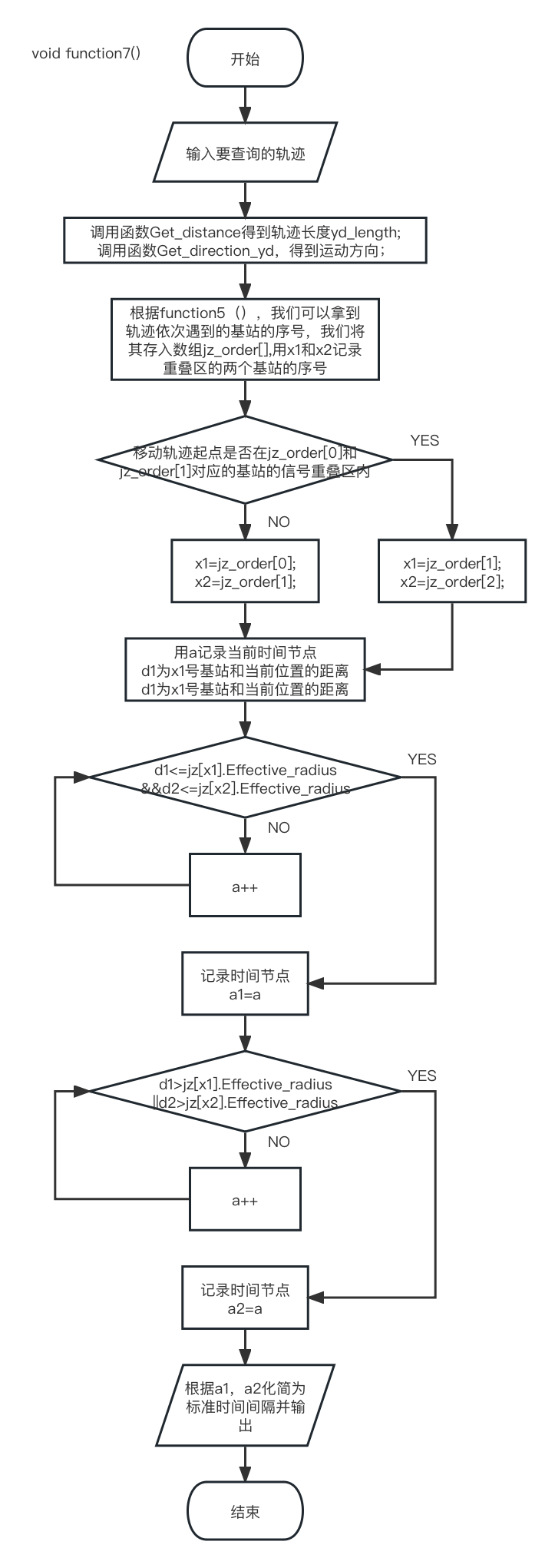


图4-2-6 function7函数流程图

根据function5，我们可以拿到轨迹依次遇到的基站的序号，我们将其存入数

组jz\_order,我们判断移动轨迹起点是否在jz\_order[0]和jz\_order[1]的重叠区，若不在，则直接找进入和离开jz\_order[0]和jz\_order[1]的重叠区的时间节点即可；若在，则要找进入和离开jz\_order[1]和jz\_order[2]的重叠区的时间节点。而进入这个临界对应的情况是：第一次终端同时在两个基站的信号范围之内。离开这个临界对应的情况是：终端离开了任意一个的信号范围。由此便可求出在覆盖区的时间长度。

8&9. 第9、12段轨迹连接上伪基站的情况（如图4-2-7所示）

任务仅要求判断第9、12段轨迹是否连接上为基站，若连接上则输出，连接上的时刻、伪基站的编号、以及连接的时间长度，但我将功能扩展为所有轨迹都可以查询该功能。用户输入查询的轨迹后，我们将运动轨迹终点到起点的连续路径进行密集的均分，用密集的离散的点代替一段连续的轨迹，同时将伪基站的运动轨迹也用密集的点代替，以时间为变量，遍历每一个时刻，查找是否有伪基站和移动终端距离第一次在40m以内，若有则记录时间节点T1，记录该伪基站的编号N，并记录下一次距离大于40m的时间节点T2，则X=T1，Y=N，T=T2-T1，输出这三个量即可。

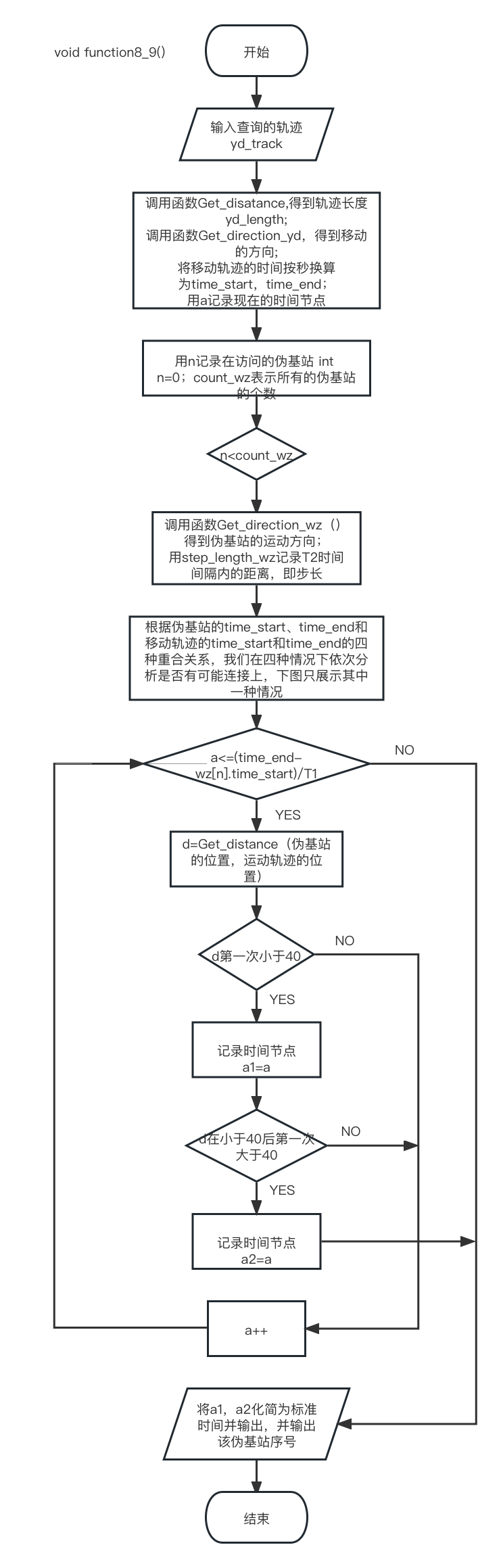


图4-2-7 function8\_9函数流程图

# 五、系统实现

## 1.开发环境

使用的开发环境是Clion。（如图5-1所示）

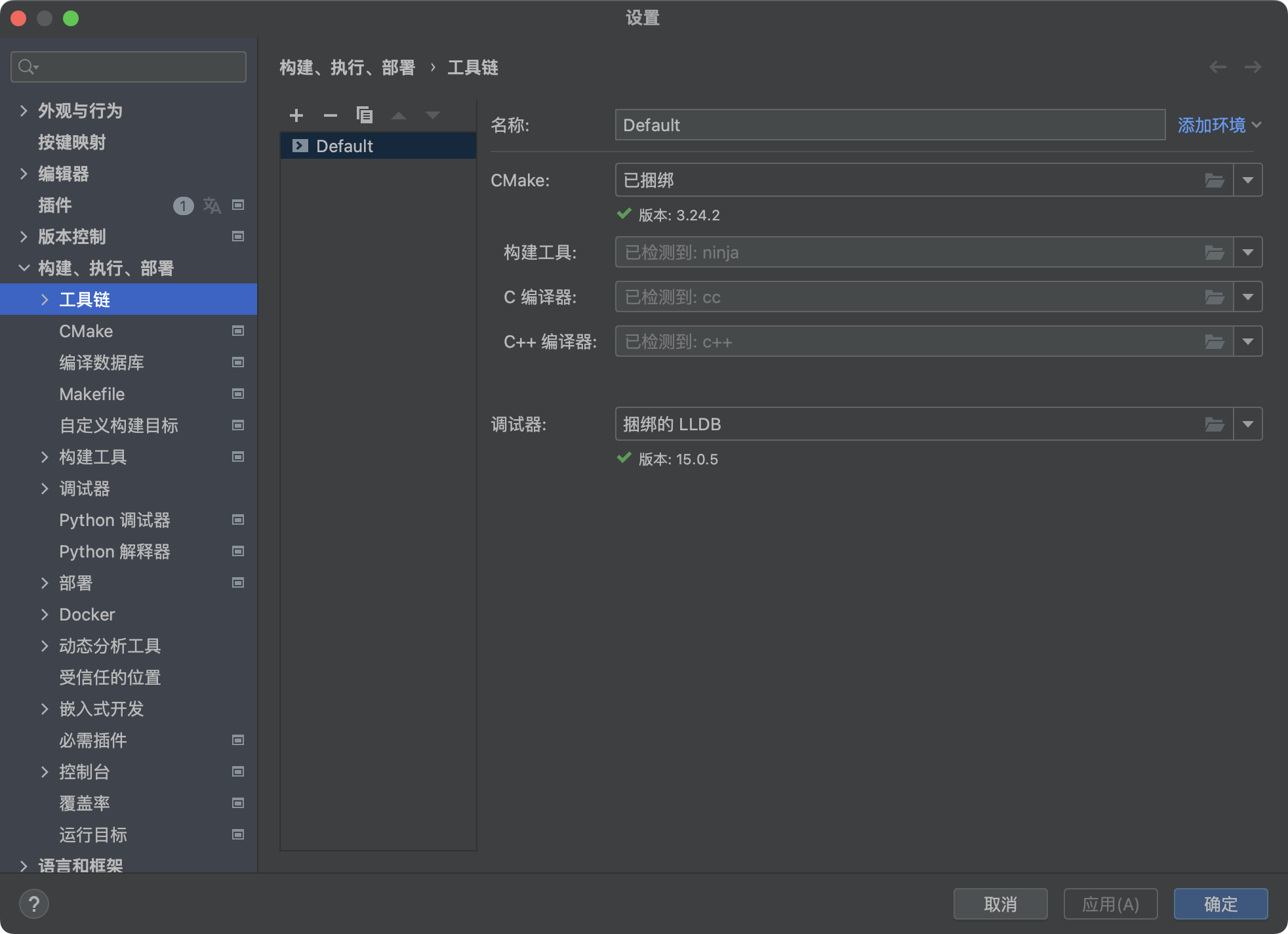


图5-1 开发环境示意图

## 2.调用的库

#include "stdio.h"

#include "string.h"

#include "math.h"

#include "stdlib.h"

## 3. 主要函数原型与功能及调用关系

（1）void Save\_JZ()

函数功能：将基站的信息读入

入口参数：无

出口参数：无

调用的函数：无

（2）void Save\_YD()

函数功能：将移动终端的移动轨迹信息读入

入口参数：无

出口参数：无

调用的函数：无

（3）void Save\_WZ()

函数功能：将伪基站的移动轨迹信息读入

入口参数：无

出口参数：无

调用的函数：无

（4）void Extreme\_value()

函数功能：获取基站信息中坐标X，Y的最值，方便确定分块的上下界

入口参数：无

出口参数：无

调用的函数：无

（5）void Partition1()

函数功能：将全部基站按照1000\*1000的大小进行分块

入口参数：无

出口参数：无

调用的函数：无

（6）void Partition2()

函数功能：将高速基站按照5000\*5000的大小进行分块

入口参数：无

出口参数：无

调用的函数：无

（7）double Get\_distance(int x1,int y1,int x2,int y2)

函数功能：获取（x1，y1），（x2，y2）两坐标间的距离

入口参数：两点坐标（x1，y1），（x2，y2）

出口参数：返回值的形式返回了两点间的距离d

调用的函数：无

（8）int Search\_max\_power(int x,int y,int X,int Y,double\* power)

函数功能：查找在（X，Y）分块中对于坐标（x，y）信号最强的基站

入口参数：点坐标（x，y），分块（X，Y）

出口参数：返回值的形式返回了最强基站的序号，power返回了最强的信号大小

调用的函数：double Get\_distance(int x1,int y1,int x2,int y2)

（9）int Get\_max\_power(double x,double y,double\*power)

函数功能：查找对于坐标（x，y）信号最强的基站

入口参数：点坐标（x，y）

出口参数：返回值的形式返回了最强基站的序号，power返回了最强的信号大小

调用的函数：int Search\_max\_power(int x,int y,int X,int Y,double\* power)

（10）void Get\_direction\_yd(double\* sin,double\* cos,int track,double yd\_length)

函数功能：获取移动终端的移动方向

入口参数：移动轨迹的编号track，轨迹的长度yd\_length

出口参数：移动的方向（cos，sin）

调用的函数：无

（11）void Get\_direction\_wz(double\* sin,double\* cos,int track,double wz\_length)

函数功能：获取伪基站的移动方向

入口参数：伪基站的编号track，轨迹的长度yd\_length

出口参数：移动的方向（cos，sin）

调用的函数：无

（12）void function1()

函数功能：实现功能1，首个和最后分块基站数据的显示

入口参数：无

出口参数：无

调用的函数：无

（13）void function2()

函数功能：实现功能2，显示特定分块的基站数据

入口参数：无

出口参数：无

调用的函数：无

（14）void function3\_4()

函数功能：实现功能3、4，判定坐标是否被覆盖，找到指定坐标附近最强信号的基站信息

入口参数：无

出口参数：无

调用的函数：int Get\_max\_power(double x,double y,double\*power)

（15）void function5()

函数功能：实现功能5，终端移动时基站的切换情况

入口参数：无

出口参数：无

调用的函数：double Get\_distance(int x1,int y1,int x2,int y2)，

void Get\_direction\_yd(double\* sin,double\* cos,int track,double yd\_length)，

int Get\_max\_power(double x,double y,double\*power)

（16）void function6()

函数功能：实现功能6，判断第1段移动轨迹进入/离开首个基站的时间信息

入口参数：无

出口参数：无

调用的函数：double Get\_distance(int x1,int y1,int x2,int y2)，

void Get\_direction\_yd(double\* sin,double\* cos,int track,double yd\_length)，

int Get\_max\_power(double x,double y,double\*power)

（17）void function7()

函数功能：实现功能7，判段第3、6段轨迹处于首个信号重叠区的的时间长度

入口参数：无

出口参数：无

调用的函数：double Get\_distance(int x1,int y1,int x2,int y2)，

void Get\_direction\_yd(double\* sin,double\* cos,int track,double yd\_length)，

int Get\_max\_power(double x,double y,double\*power)

（18）void function8\_9()

函数功能：实现功能8、9，查看第9、12段轨迹连接上伪基站的情况

入口参数：无

出口参数：无

调用的函数：double Get\_distance(int x1,int y1,int x2,int y2)，

void Get\_direction\_yd(double\* sin,double\* cos,int track,double yd\_length)，

void Get\_direction\_wz(double\* sin,double\* cos,int track,double wz\_length)

# 六、运行测试与结果分析

1.预备信息提醒（如图6-1所示）

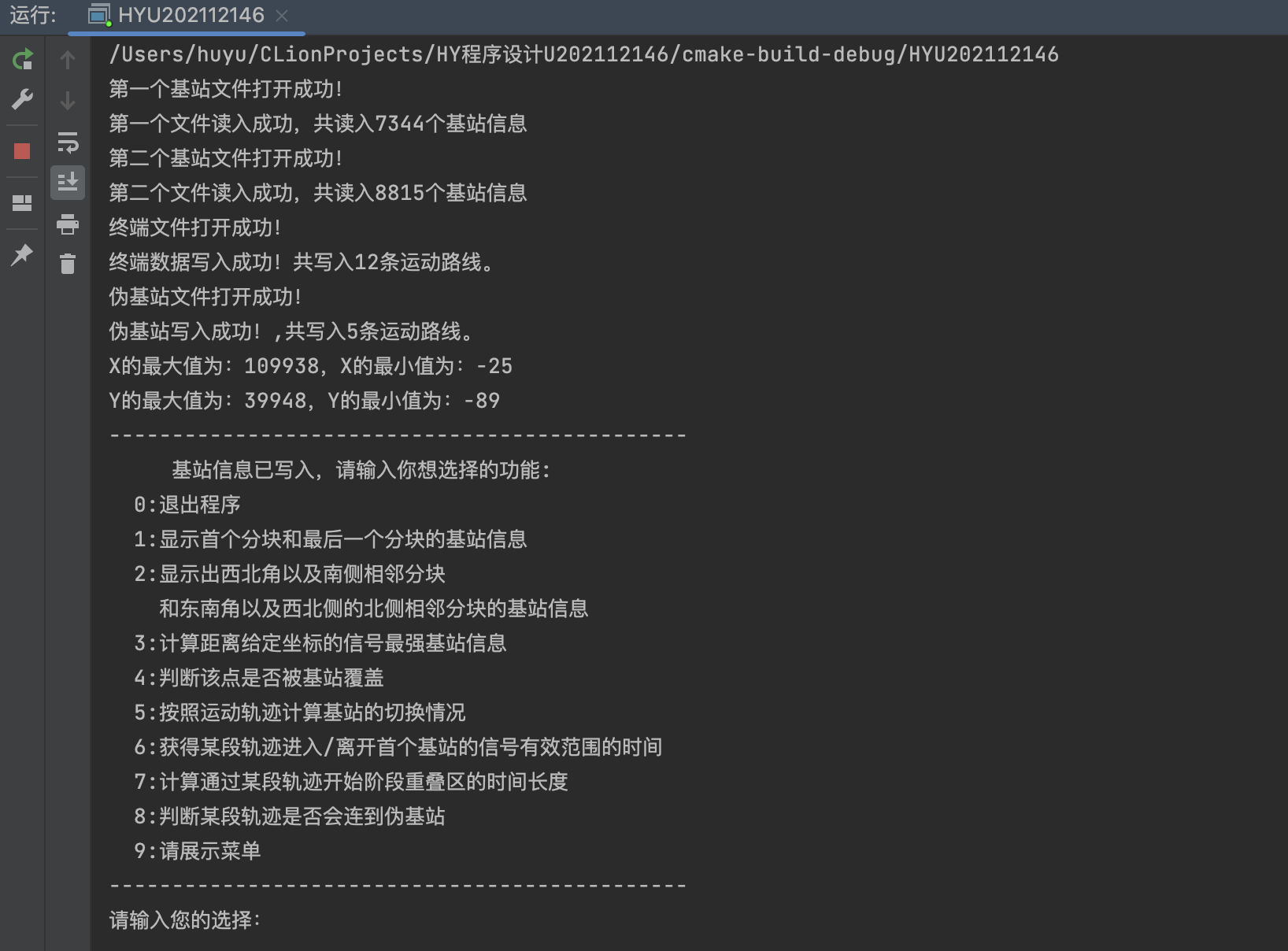


图6-1 预备信息提醒示意图

2.功能1的实现（如图6-2所示）

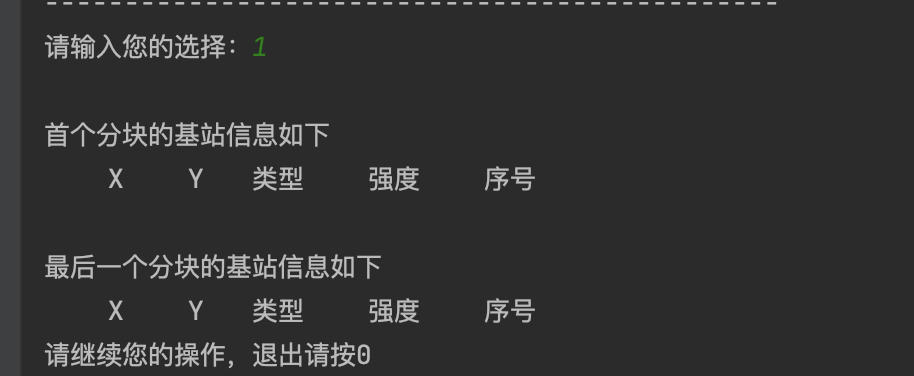


图6-2 功能1实现示意图

由于分块显示，和块的大小有极大关系，特此说明，这是按照1000\*1000的显示结果。

3.功能2的实现（如图6-3所示）

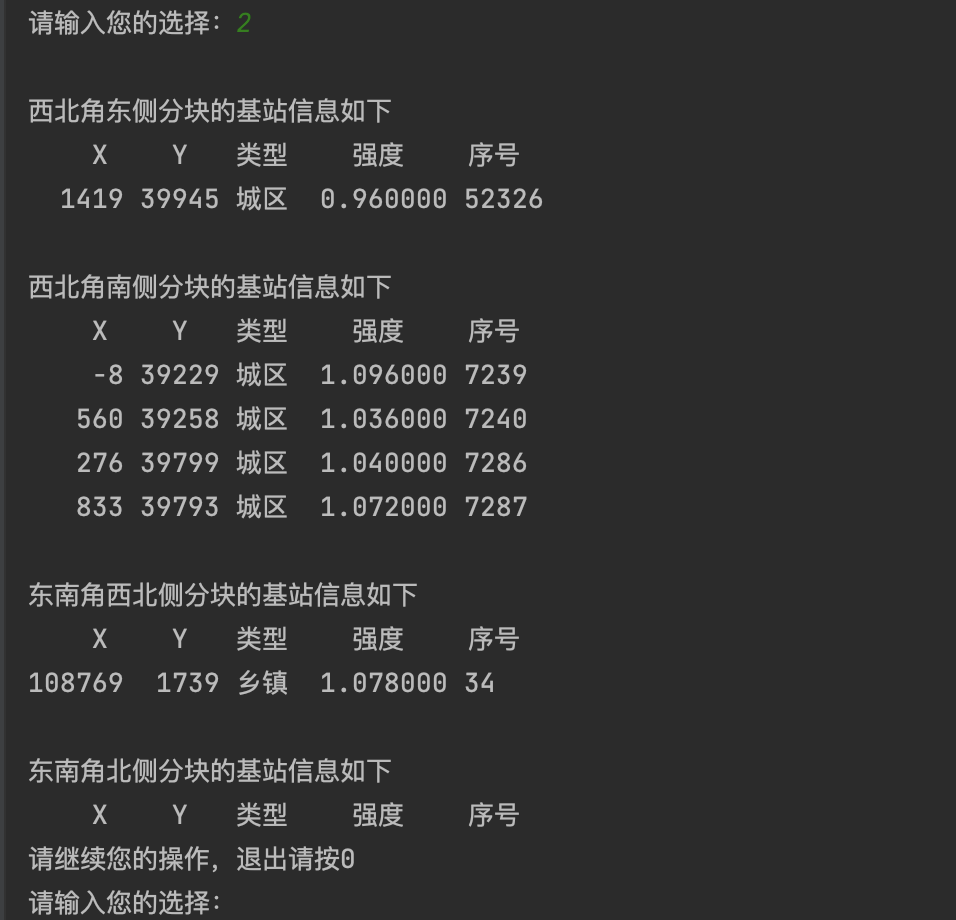


图6-3 功能2实现示意图

由于分块显示，和块的大小有极大关系，特此说明，这是按照1000\*1000的显示结果。

4.功能3的实现（如图6-4所示）

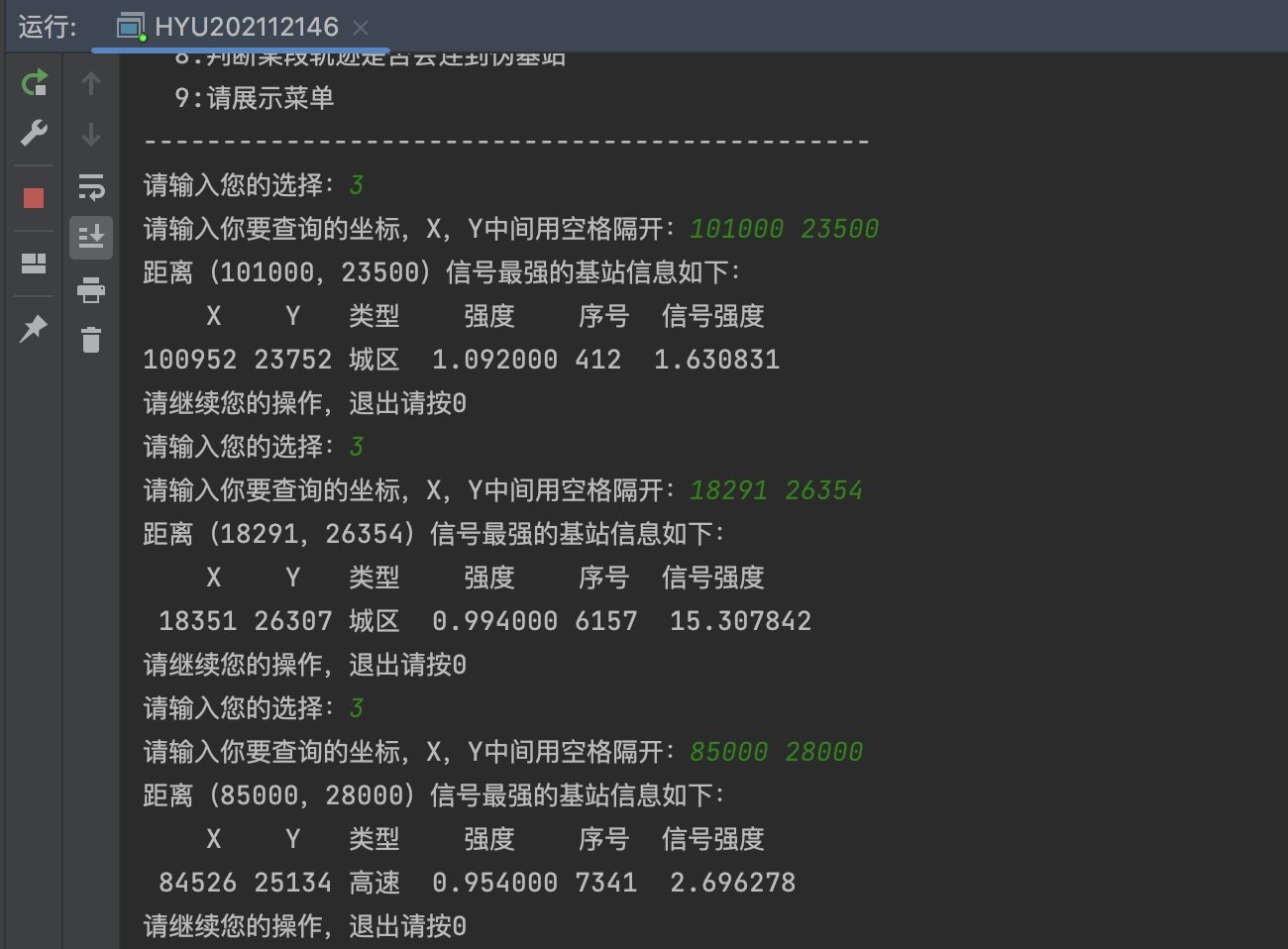


图6-4 功能3实现示意图

5.功能4的实现（如图6-5所示）



图6-5 功能4实现示意图

6. 功能5的实现（如图6-6，6-7，6-8，6-9所示）

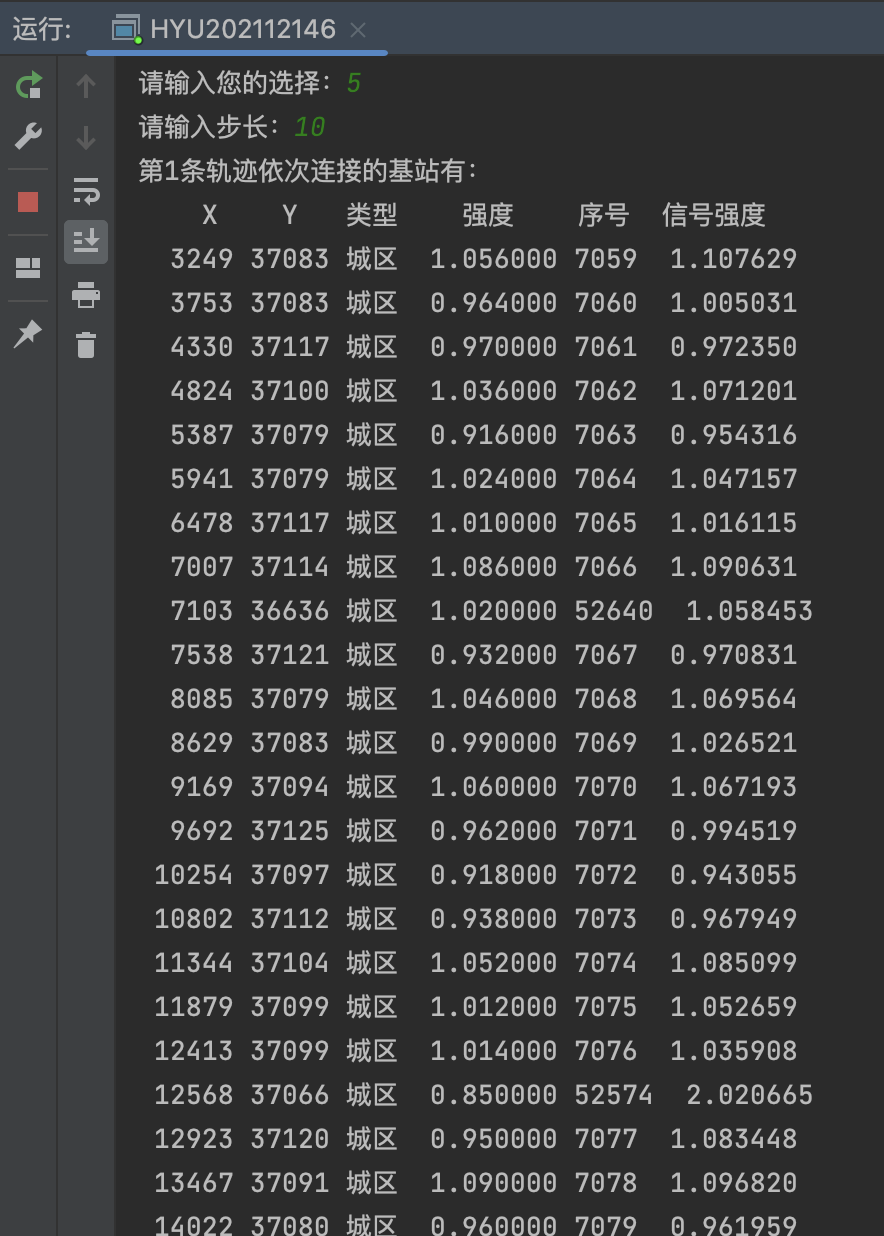


图6-6 功能5实现示意图

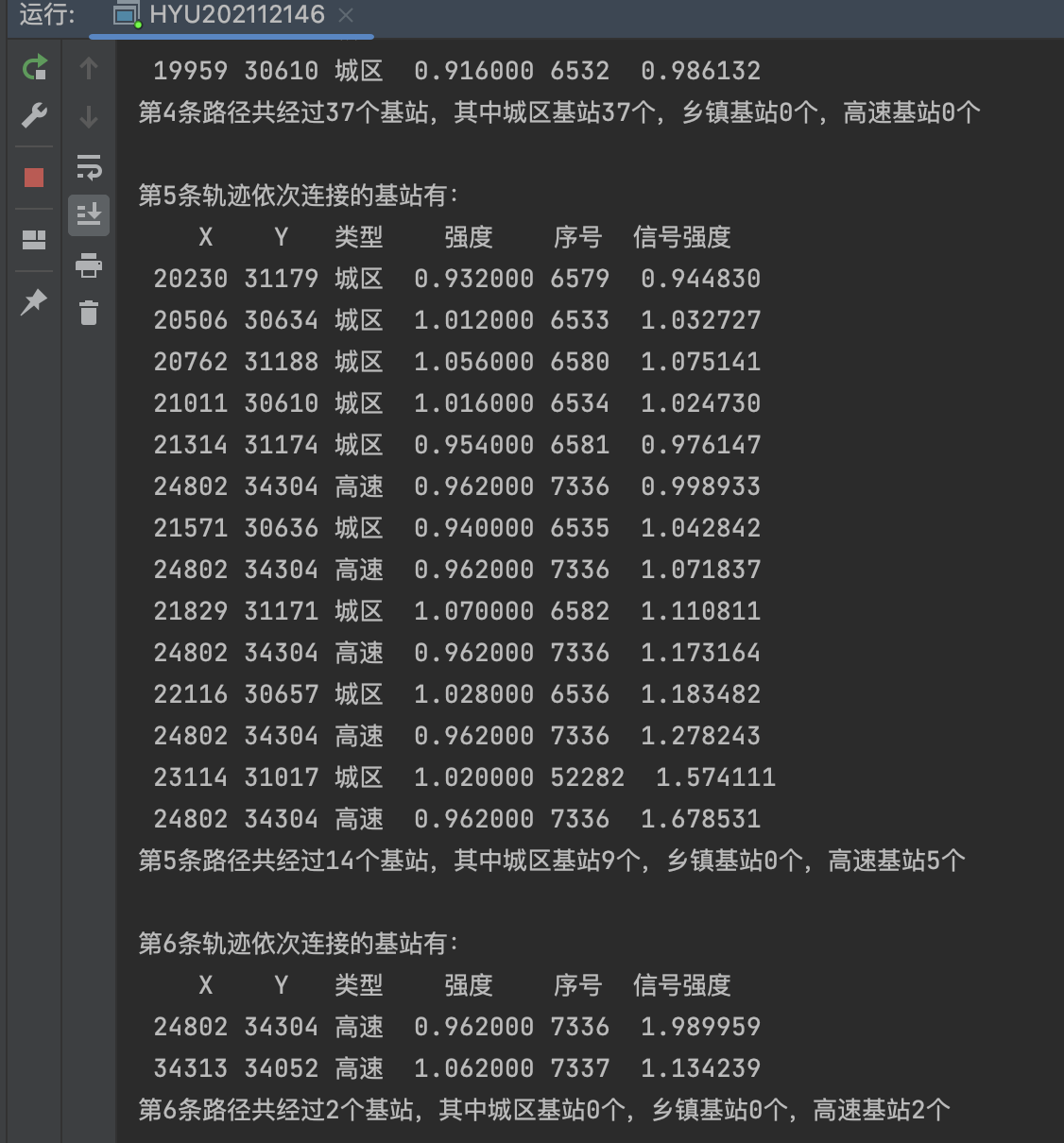


图6-7 功能5实现示意图



图6-8 功能5实现示意图

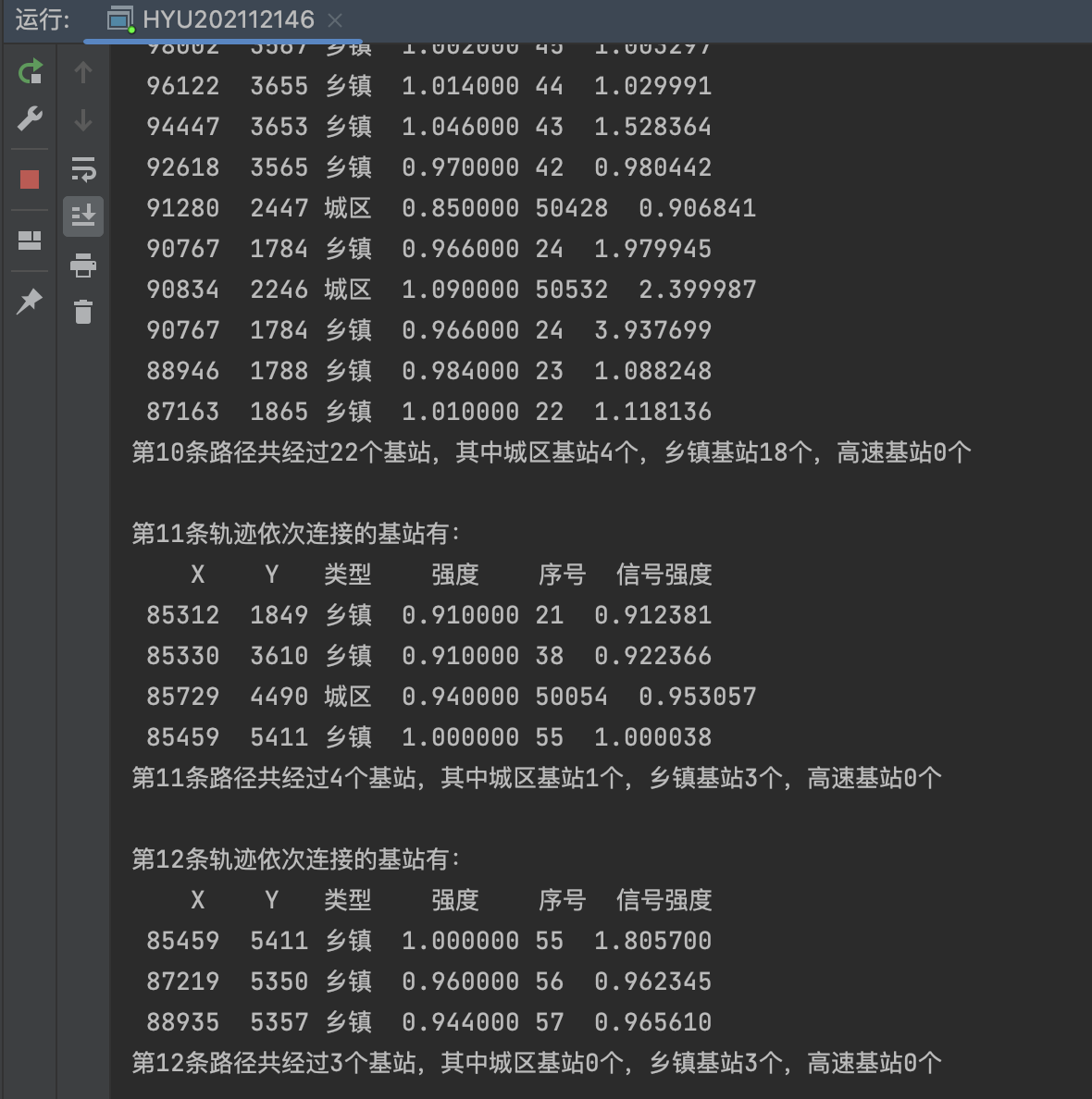


图6-9 功能5实现示意图

功能5的输出过多，这里仅展示部分有代表性输出。

7.功能6的实现（如图6-10所示）

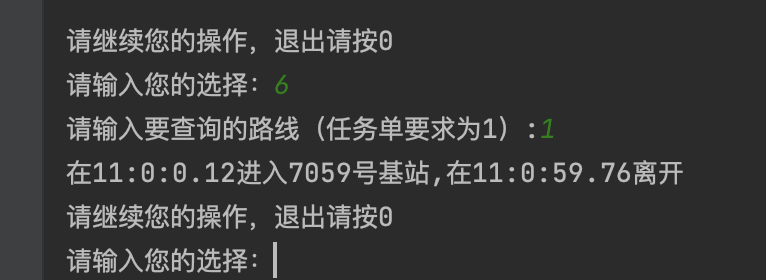


图6-10 功能6实现示意图

8.功能7的实现（如图6-11）

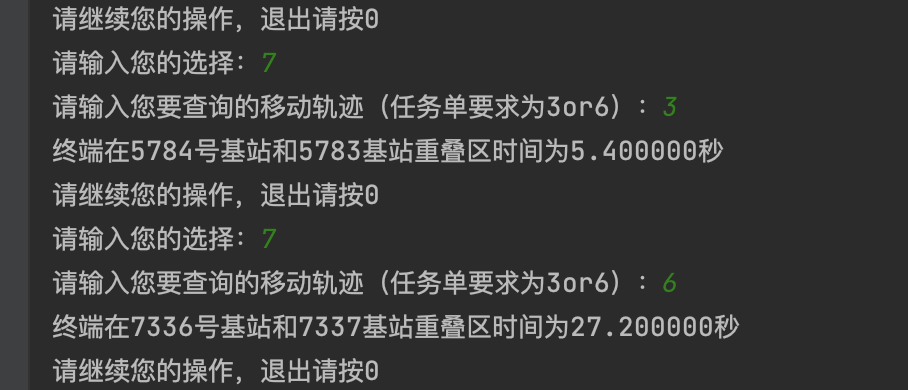


图6-11 功能7实现示意图

8.功能8的实现（如图6-12）

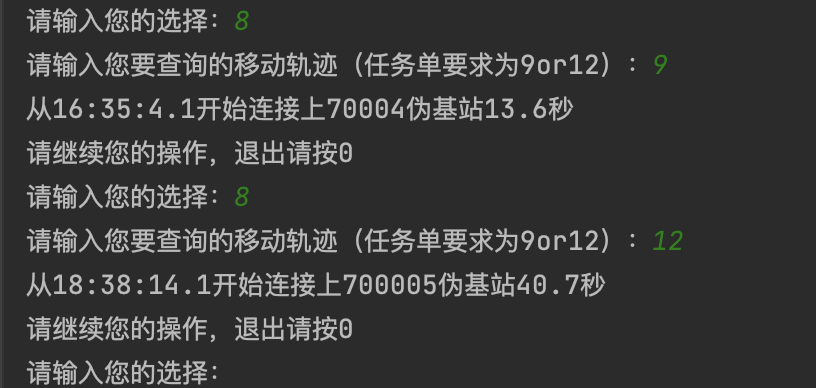


图6-12 功能8实现示意图

# 七、总结

经过这次综合的课程设计，我受益良多，感慨万分，过程无疑是艰苦的，遇到了一些棘手的问题，但也锻炼了我的意志，增强的我的能力，培养了我独立自主编程的能力，每行代码都是出自我手，核心思想全是我自己一人想出的，下面容我仔细分享我的感悟。

首先，拿到问题就感觉这项任务不一般，以前从未写过大的项目，顶多是刷刷leetcode的小题，而这次课设却是一个很偏向落地的现实问题，了解到通信网络的发展，我深知身上的重担，前辈们创造了以5G和千兆光网为代表的‘双千兆’网络，具有超大带宽、超低时延、先进可靠等特征，而这些是新型基础设施的重要组成和承载底座，现在时代的接力棒即将交到我们青少年的手上，我一定不辱使命，学成之时，建设祖国。

在进行这项大的项目编程的过程中，我深深体悟到了它和那些小的算法题的不同，明白了流程图、框架图的意义。我们应先聚焦于需要实现的功能，搭建一个大的体系，进行模块化编程，先明确每个模块的作用，通过框架图明晰每个模块所要实现的功能。我们还需要关注于我们要处理的数据，根据数据本身，我们要选择合适的数据结构，定义合适的结构体。然后我们才在不同的模块中去关注每个具体功能的实现，而有些功能本身其实是有共性的，是相关联的，我们可以将这些共性封装成一些基本的函数，以便于主体功能的实现，而对于主体功能的实现函数，我希望它封装的尽可能简洁，让其通俗易懂。

遇到的问题更是数不胜数，下面容我一一道来，我纠结过是选择四叉树还是分块来处理这个问题，看到主要功能一二，要求输出基站信息，而四叉树的块的大小是不一样的，不太好输出，于是我选择了分块的处理方法。在分块时，又存在两个问题，一是基站坐标总的范围如何确定？二是分多大的块比较合适？对于问题一，我考虑到程序的可移植性，我将总的范围定为基站数据中的坐标最小值到最大值，而并不是某个确定的值。对于问题二，我经过对三种基站数据的观察，我发现高速的信号有效范围远大于城区和乡镇，于是我决定按5000\*5000单独将高速分块，而按1000\*1000将所有基站分块，两种分块方式的好处是，在查找信号最强的基站时，遍历的基站数会大大减少，而且能确保结果的可靠性，减少了算法的时间复杂度。在实现功能3、4时，我发现我需要遍历坐标所在附近的9个分块，查找信号最强的基站，如果将其写在一个函数中未免太过冗杂，于是我将查找一个分块中信号最强的基站封装成一个函数Search\_max\_power，而我的查找信号最强的基站的函数Get\_max\_power只需调用9次Search\_max\_power就好，这就是模块化设计和封装的好处！在实现扩展功能2时，其实也遇到过问题，有一个很坑的点，就是轨迹的起点就在两个基站信号的重叠区，那么此时应该是不算数的，我们应遍历到下两个基站，去计算后两个基站重叠区的进入和离开时间节点，这点我先没有考虑到，在用数据2测试时，就出了问题，好在后面助教提醒后我及时进行了改进。再就是在升级功能的实现时，这个算法我思考了小半天，对我来说属于一个小难点了，最后我也算是通过动手画图解决了这个问题，首先，其实文件中给的时间数据是不好直接用的，我们应先将移动终端的时间数据和伪基站移动轨迹的时间数据全部换算成以秒为单位，这样会极大程度的方便进行比较。然后我们应该遍历所有的伪基站移动轨迹，找出其时间上和终端的重合，再在重合的时间中去考虑距离是否小于40，需要注意的是，时间重合的情况实际上是有4种情况的，报告中已经指出，这里就不多赘述了，于是这个问题就完美的解决了。

在思维上，我最大的感悟就是工程思维，面对一项工程，如何设计模块，如何设计数据机构，在哪些模块中实现哪些功能，哪些功能的利用次数很多，我们因将其封装成基本函数，这些才是重要的地方，最后才是代码实现。而且我们的程序最后是面向用户的，我们也要从用户的体验角度去考虑问题，于是我就在代码中加入了菜单模块，感觉效果还是不错的！

最后，希望自己吸取教训，虚心求学，继续努力，不负尊长所付，当秉青云之志！

# 八、参考文献

[1]严蔚敏.数据结构C语言版[M].清华大学出版社，2007

[2]C Primer Plus中文版(第6版) ,人民邮电出版社出版，英文原版书名：C Primer Plus Sixth Edition。作者：蒂芬•普拉达 (Stephen Prata) 著, 张海龙, 袁国忠 译