

东软秘密

**设计报告  
*（电子地图信息统计系统）***

东软集团股份有限公司IT人才实训中心

版权所有，翻版必究

**更改履历**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 版本 | 更改时间 | 更改人 | 更改章节 | 状态 | 更改描述 |
| 1 | V1.0 | 2019-7-17 | 马振乐 | 主函数、头文件模块 | 新建 | 编写建立主函数、头文件模块 |
| 2 | V1.0 | 2019-7-19 | 马振乐 | 读写模块 | 新建 | 编写建立读写模块 |
| 3 | V1.0 | 2019-7-20 | 马振乐 | 排序模块 | 新建 | 编写建立排序模块 |
| 4 | V1.0 | 2019-7-20 | 马振乐 | 检索模块 | 新建 | 编写建立检索模块 |

状态：新建、增加、修改、删除。

目 录

[1 引言 3](#_Toc224533617)

[1.1 编制目的 3](#_Toc224533618)

[1.2 词汇表 3](#_Toc224533619)

[1.3 参考资料 3](#_Toc224533620)

[2 系统开发环境 4](#_Toc224533621)

[3 系统设计思路 4](#_Toc224533622)

[4 系统公用头文件 4](#_Toc224533623)

[5 功能模块设计 5](#_Toc224533628)

[6 运行测试 6](#_Toc224533628)

# **引言**

## 编制目的

本报告详细完成对*电子地图信息统计*系统的整体设计，达到指导开发的目的，同时实现和测试人员及用户的沟通。

本报告面向详细设计人员、开发人员、测试人员及最终用户而编写，是了解系统的导航。

## 词汇表

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| 大端 | 是指数据的高位，保存在内存的低地址中，而数据的低位，保存在内存的高地址中， | 本系统没有用到该知识点 |
| 小端 | 是指数据的高位保存在内存的高地址中，而数据的低位保存在内存的低地址中 | 本系统有用到该知识点 |
| 快速排序 | 首先任意选取一个数据（通常选用数组的第一个数）作为关键数据，然后将所有比它小的数都放到它左边，所有比它大的数都放到它右边，这个过程称为一趟快速排序。 | 排序速度快，一般用于无序的  序列中 |
| 折半查找 | 二分查找也称折半查找（Binary Search），它是一种效率较高的查找方法。但是，折半查找要求线性表必须采用顺序存储结构，而且表中元素按关键字有序排列。 | 必须采用[顺序存储结构](https://baike.baidu.com/item/%E9%A1%BA%E5%BA%8F%E5%AD%98%E5%82%A8%E7%BB%93%E6%9E%84" \t "_blank)和按关键字大小有序排列 |
| 结构体数组 | 用于定义多个成员变量，存储多个不同数据类型的数据信息 | 使用方便，经常使用 |

## 参考资料

老师所发的各种相关材料文件及项目要求。

# **系统开发环境**

操作系统： *Windows 10*

集成开发工具：*CodeBlocks*

编译环境： C/C++

Web服务器：无

# **系统设计思路**

**分层架构:**主要分为四大模块

1. 首先是头文件模块，此模块中主要用于定义宏文件、定义结构体数组和定义自己所用到的函数；
2. 然后是读写二进制文件的模块，此模块主要功能是对二进制文件进行读取，把读取到的数据反转后写入目标文本文件中，同时把道路的数据信息存储到结构体数组中；
3. 接着是道路信息排序模块，此模块的主要功能是把已经存储到结构体数组的数据进行排序，并且把排序后的结果写到另一个文本文件中；
4. 最后是道路信息查找的模块，此模块的功能是在数据已完成排序的基础上，对道路信息的检索。

**框架：**主要包括头文件模块、读写二进制文件的模块、道路信息排序模块和道路信息查找的模块

**关键性技术：**

1. 整个系统的设计主要围绕一个全局变量的结构体数组来展开设计，首先进行二进制文件的读取，让结构体数组中存储读取到的所有道路信息，然后接下来的排序和检索都利用此结构体；
2. 由于在读取二进制文件时是用大端模式的，读到的数据时逆转的，所以要定义宏来反转读取到的数据。

# **系统公用头文件**

**1**．**用到的系统头文件：**

#include <stdio.h> //输入输出标准流

#include<io.h> //定义一些和缓冲区相关的读写函数

#include<iostream> //C++输入输出标准流

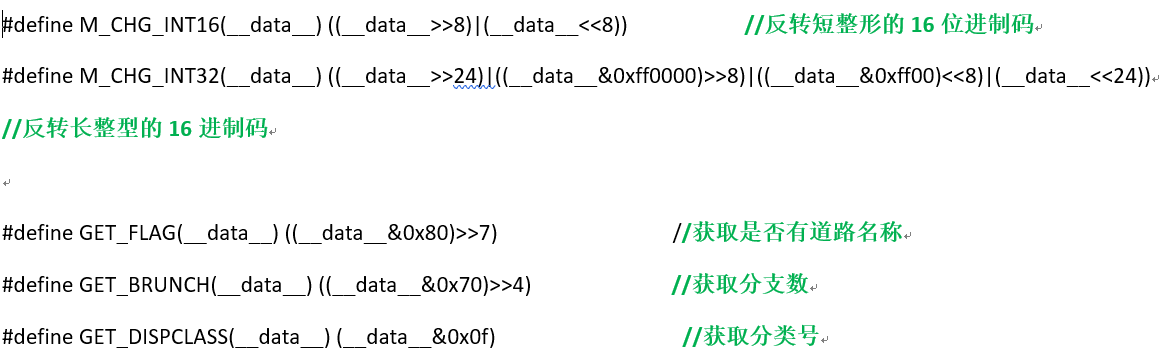
#include<cstdlib> //https://blog.csdn.net/Iaired/article/details/78700180

#include<string.h> //字符串库函数

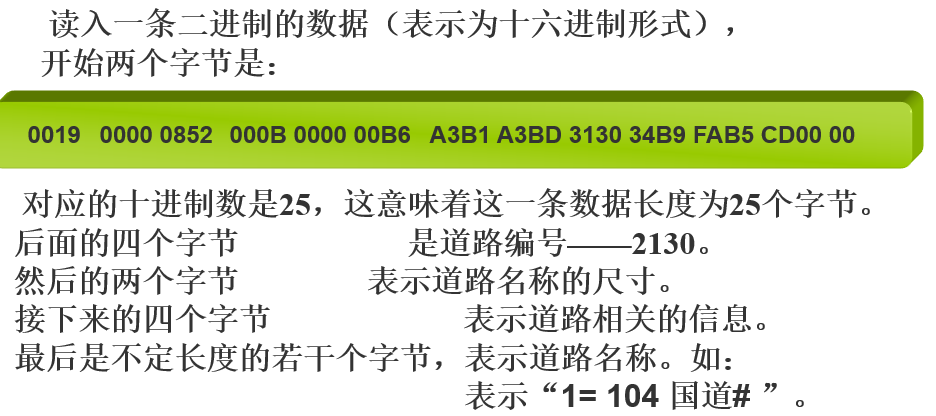
#include”header.h” //各个模块都需引用此头文件

**2.宏反转：（处理小端数据）：**

**下图显示的是宏反转的宏文件，因为读取二进制文件后的数据数逆转的，所以要接借助宏来反转各种数据类型得数据，从而得到我们想要的数据信息。**



**3．关于各个数据类型字节的分配**：



**unsigned short int recordSize—— 记录的大小（2个字节）**

**unsigned int LinkID—————— 道路的ID(4个字节)**

**unsigned int roadInfo ————— 道路信息（4个字节）**

**unsigned short roadNameLen—— 道路名字的长度（2个字节）**

**char roadName[30] —————— 道路名（13个字节）**

**4.头文件的主要代码及其功能：**

#ifndef HEADER\_H\_INCLUDED //重复包含（重复定义）

#define HEADER\_H\_INCLUDED

#define MAXSIZE 64000 //宏定义数组的最大值的MAXSIZEtypedef struct RoadRecord //结构体RoadRecord（道路记录）存储路的8个信息（8个成员变量）

{

unsigned short int recordSize; //记录的大小（2个字节）

unsigned int LinkID; //道路的ID(4个字节)

unsigned int roadInfo; //道路信息（4个字节）

unsigned short roadNameLen; //道路名字的长度（2个字节）

char roadName[30]; //数组存储道路名，只运行名字长度不能30个（其余的

//的字节全给道路名）

unsigned int disclass; //番号信息

unsigned int brunch; //分支

int flag; //用flag表示道路是否有名称（1代表有，0代表无）

}RoadRecord;

typedef struct ElecMap //结构体ElecMap存储全部道路的信息（3个成员变量），包括道路记录信息，路的数量和道路是否排序好的标志

{

RoadRecord roadRecord[MAXSIZE]; //结构体数组存储道路的记录信息，而且数组的最大存储空间为6400

int roadNum,roadSort; //roadNum表示道路的数量；roadSort表示是否排序好（1代表已排序，0代表未排序）

}ElecMap;

#define M\_CHG\_INT16(\_\_data\_\_) ((\_\_data\_\_>>8)|(\_\_data\_\_<<8)) //反转短整形的16位进制码

#define M\_CHG\_INT32(\_\_data\_\_) ((\_\_data\_\_>>24)|((\_\_data\_\_&0xff0000)>>8)|((\_\_data\_\_&0xff00)<<8)|(\_\_data\_\_<<24))

//反转长整型的16进制码

#define GET\_FLAG(\_\_data\_\_) ((\_\_data\_\_&0x80)>>7) //获取是否有道路名称

#define GET\_BRUNCH(\_\_data\_\_) ((\_\_data\_\_&0x70)>>4) //获取分支数

#define GET\_DISPCLASS(\_\_data\_\_) (\_\_data\_\_&0x0f) //获取分类号

#endif HEADER\_H\_INCLUDED

# **功能模块设计**

1. **读写模块**：

功能概要：进行读取文件操作，并将读取信息存储在结构体数组中。

详细说明:1.打开并读取二进制文件 2.读取并宏反转recordSize、LinkID 、roadNameLen、roadInfo 3.在roadInfo中获取Class番号、岔路数、 flag 4.把读取和获取到的数据写入文本文件，并存储在结构体数组中5.判断roadSiae-12是否大于0, 若大于0读取道路名称并存储， 否则输出无道路名称。

int ReadFile(ElecMap \*eMap); //读取二进制文件，并转换存储在文本文档中

1. **排序模块**

功能概要：对存储进去结构体数组的数据信息用各种排序方法进行排序。

详细说明：1.新建并打开要存储排序结果的文本文件2.用switch开关语句选择排序方式(排序方式包括快速排序、冒泡排序、直接插入排序、简单选择排序、)3.把排序后的结果存储进去文本文件4.分析和比较各种排序方法的性能。

int TotalSort(ElecMap \*eMap); //排序

int Sort(ElecMap \*eMap); //快速排序

int QSort(ElecMap \*eMap,int left,int high);

int Partition(ElecMap \*eMap,int left,int high); //进行一趟排序，找到并返回枢轴位置

int BubbleSort(ElecMap \*eMap); //冒泡排序

int DirectInsertSort(ElecMap \*eMap); //直接插入排序

int BinarySort(ElecMap \*eMap); //二分排序

int SimpleElectSort(ElecMap \*eMap); //简单选择排序

int ToWriteSort(ElecMap \*eMap); //把排序好的道路信息写入文本文档中

1. 检索模块

功能概要:在结构数组已完成排序的基础上，对道路信息数据进行检索。

详细说明:1.新建并打开来存储不同方式检索结果对应的目标文本文件2.用switch开关语句选择检索方式(包括LinkID检索、道路名检索、道路分支检索、番号检索)3.把检索结果大于5条的结果存储在对应目标文本文件中，方便查阅；把检索结果小于5条的结果直接输出显示屏幕中。

int Search(ElecMap \*eMap); //检索

int BinarySearch\_LinkID(ElecMap \*eMap); //用二叉排序对道路LinkID进行检索

int LinearSearch\_RoadName(ElecMap \*eMap); //用直接插入排序对路名进行检索

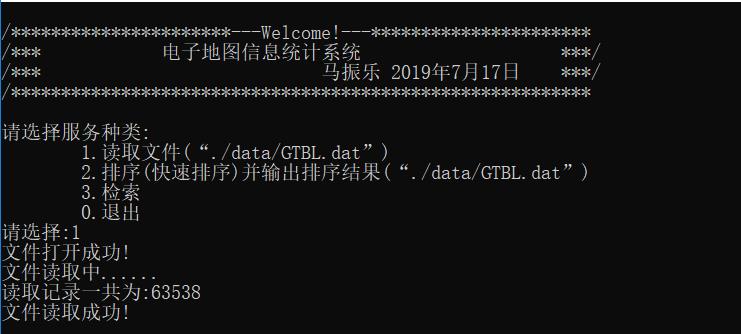
int LinearSearch\_RoadBrunchNum(ElecMap \*eMap,int key); //用直接插入排序对道路分支数进行检索

int LinearSearch\_Class(ElecMap \*eMap,int key); //用直接插入排序对番号进行检索

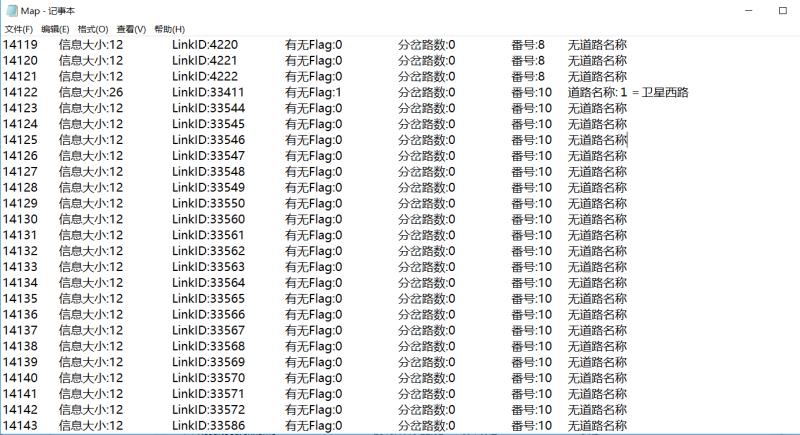
# **系统运行测试**

1. **系统启动及读取文件：**

系统运行结果：

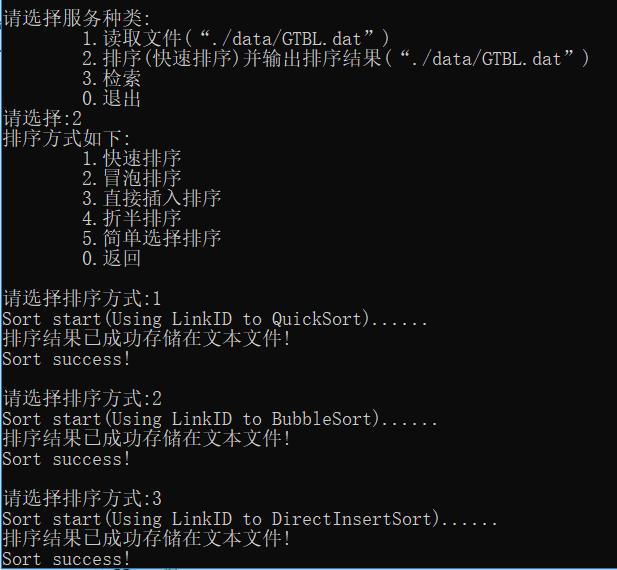


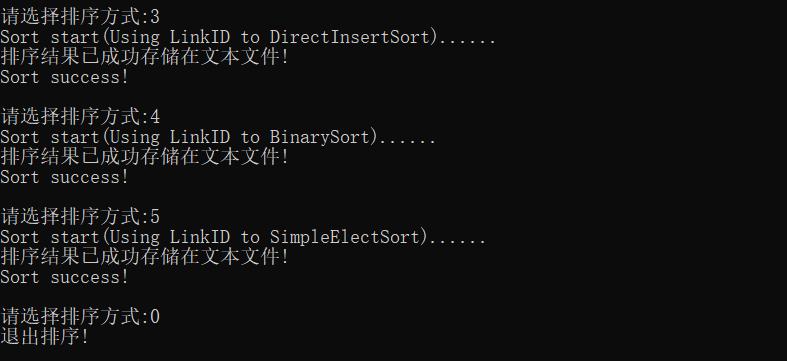
写入文本文件的结果：



1. **道路信息数据的排序：**

系统运行结果：



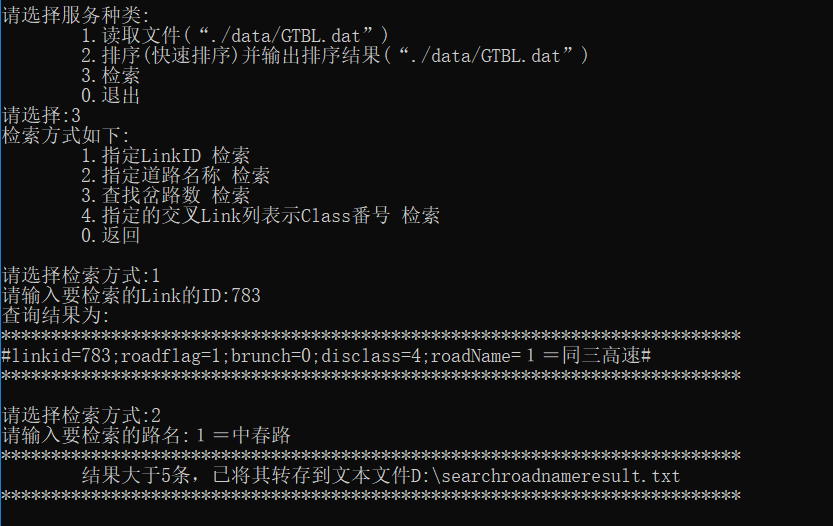


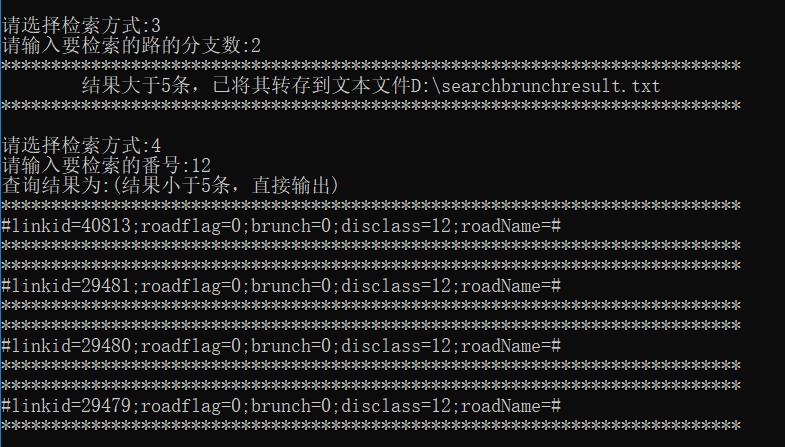
写入文本文件的结果:

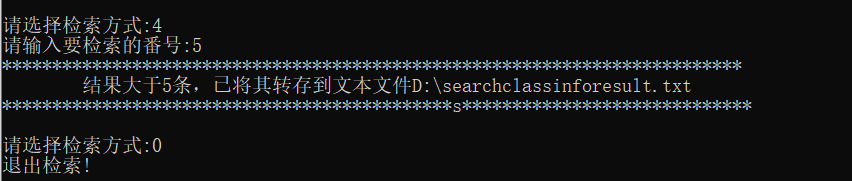


1. **道路信息数据的检索：**

系统运行结果：

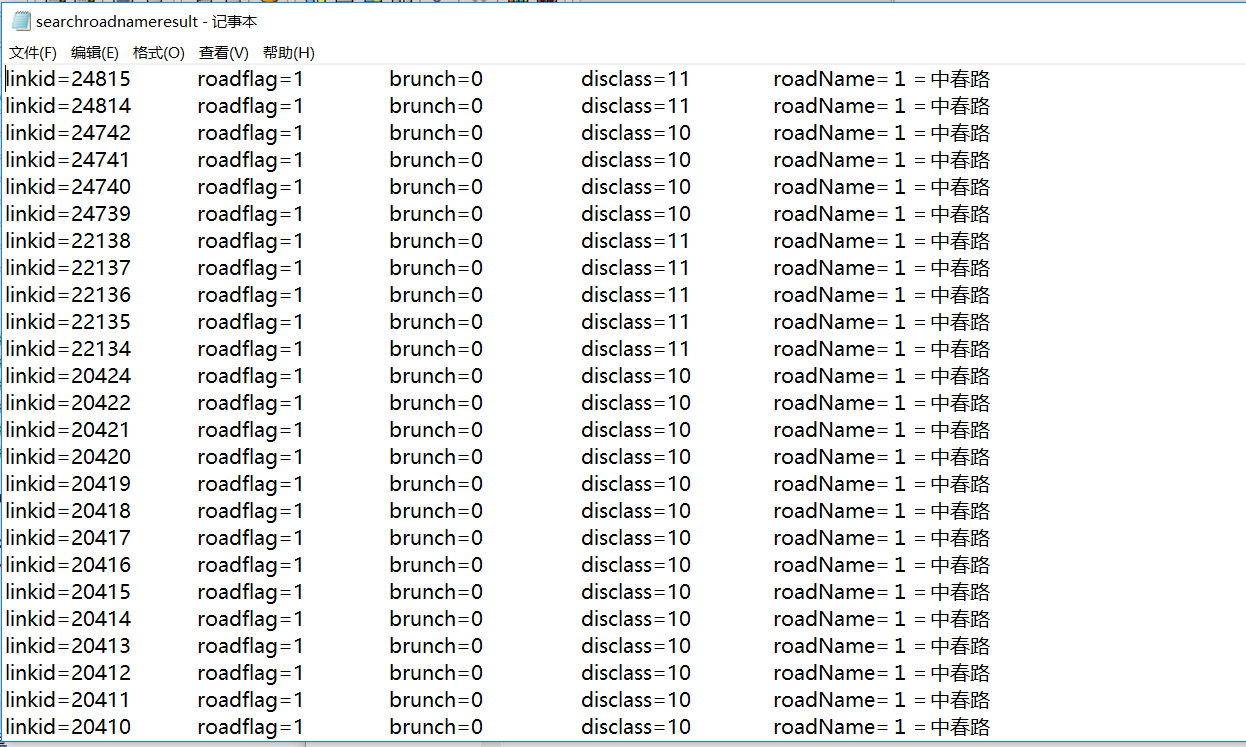






写入文本文件结果:

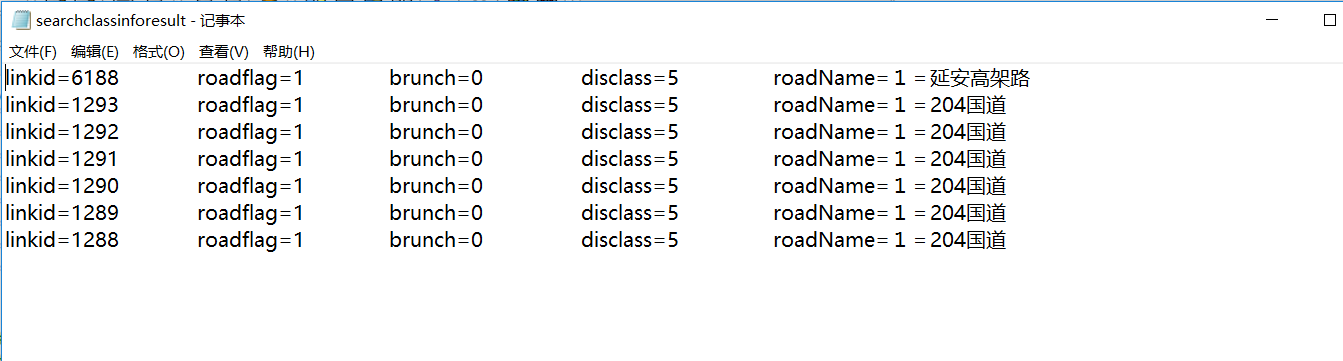
用道路名称检索写入文本文件结果:



用分支数检索写入文本文件的结果:



用番号检索写入文本文件的结果:



1. **退出系统：**

