**数据结构课程设计**

**班级：1615403**

**学号：161540205**

**姓名：张佳钰**

**指导老师：孙涵**

5、Hash表应用 （必做） （查找）

[问题描述]

设计散列表实现身份证查找系统，对身份证号进行Hash。

[基本要求]

（1） 设每个记录有下列数据项：身份证号码（虚构，位数和编码规则与真实一致即可）、姓名、地址；

（2） 从键盘或文件输入各记录，至少50个以上，以身份证号码为关键字建立散列表；

（3） 采用链地址的方法解决冲突；

（4） 查找并显示给定身份信息的记录；

采用的数据结构：

链表

算法思想：

通过连地址解决冲突

源程序：

/\*

53 012 m 320602198609292020

54 102 w 3206231987081527508

55 110 m 320611198611252614

56 111 w 320602198606300021

57 112 w 320602198509120547

60 113 m 320683198511032018

61 114 w 320683198812046026

\*/

//id\_example.txt

/\*

123123123

123123123

123123123

320602198704121529

320611198407212631

320623198502280013

320681198609114414

320682198505083605

32060219861072029

320621198806242048

320611198611252614

320602198606300021

320602198509120547

320683198511032018

320683198812046026

\*/

//exam.txt

#include<iostream>

#include<fstream>

using namespace std;

#include<string>

typedef struct id\_node{

int num;

string id\_num;

string name;

id\_node \*next\_id;

string sex;

id\_node()

{

sex="nu";

id\_num=' ';

name=' ';

next\_id=NULL;

}

}id\_node;

typedef struct id\_info

{

id\_node id[100];

int id\_num;

id\_info(){id\_num=0;}

}id\_all\_info;

int Hash(string a)

{

//320602198605140521

//012345678901234567

// 02 8605140521

//028+506+140+125

int b[19];

for(int i=0;i<a.length();i++)

{

b[i]=a[i]-'0';

// cout<<b[i]<<endl;

}

return ((b[4]+b[11]+b[12]+b[18])\*100+(b[5]+b[10]+b[13]+b[17])\*10+(b[8]+b[9]+b[15]+b[16]))%100;

}

void read\_id\_info(id\_all\_info &ID)

{

fstream f1;

f1.open("id\_example.txt",ios::in);

id\_node \*id,\*p;

int j;

while(!f1.eof())

{

id = new id\_node;

f1>>id->num>>id->name>>id->sex>>id->id\_num;

j=Hash(id->id\_num);

// cout<<j<<endl;

// system("pause");

ID.id\_num++;

if(ID.id[j].next\_id==NULL)

{

ID.id[j].next\_id=id;

}

else

{

p=ID.id[j].next\_id;

while(p->next\_id!=NULL)

{

p=p->next\_id;

}

p->next\_id=id;

}

}

f1.close();

}

void display\_id\_info(id\_all\_info ID)

{

int n,i=0;

n=ID.id\_num;

id\_node \*p;

while(i<100)

{

p=ID.id[i].next\_id;

while(p)

{

// system("pause");

cout<<i<<" "<<p->num<<" "<<p->name<<" "<<p->id\_num<<endl;

p=p->next\_id;

}

i++;

// cout<<i<<endl;

}

}

id\_node\* search\_is\_id\_exist(id\_all\_info ID,string a)

{

int j;

j=Hash(a);

id\_node \*p;

p=ID.id[j].next\_id;

int flag=0;

while(p&&(p->id\_num.compare(a)))

{

p=p->next\_id;

}

if(p)

{

return p;

}

return NULL;

}

void test\_search(id\_all\_info ID)

{

fstream f1;

f1.open("exam.txt",ios::in);

string b;

id\_node \*p;

int flag=0;

while(!f1.eof())

{

// system("pause");

p=NULL;

f1>>b;

if(b.length()<=20&&b.length()>=16)

{

p=search\_is\_id\_exist(ID,b);

}

else

{

cout<<"searching: "<<b<<endl;

cout<<"This id number is illegal"<<endl<<endl;

}

if(p)

{

cout<<"searching: "<<b<<endl;

cout<<p->num<<" "<<p->name<<" "<<p->sex<<" "<<p->id\_num<<endl<<endl;

}

}

f1.close();

}

int main()

{

id\_all\_info ID;

read\_id\_info(ID);

display\_id\_info(ID);

test\_search(ID);

return 0;

}

测试数据：

id\_example.txt

13 崔晓进 男 320621198703014317

14 朱玉石 女 320681198512182429

15 徐俊妹 女 320682198602161409

16 江艳 女 320602198704121529

17 黄卓 男 320611198407212631

18 吴瑶 男 320623198502280013

19 顾天华 男 320681198609114414

20 朱锦芳 女 320682198505083605

21 张慧 女 32060219861072029

22 范慧 女 320621198806242048

23 田磊 男 320602198610072003

24 王静 女 320602198607175322

25 徐晓雯 女 320602198810312521

26 袁海霞 女 320602198302030042

27 孙琴 女 320705198702051525

28 周亿文 女 320623198512217843

29 龚庆子 男 320602198605140521

30 王盈盈 女 320322198610176843

31 周琦 男 320602198702086512

32 洪燕 女 320682199004194686

33 张玲玲 女 320683198803306260

34 朱琦 女 320602198612262527

35 陈思思 女 320602198710060021

36 范丹丹 男 320683198605016721

37 黄霞 女 320682198802285481

38 徐晶晶 女 320683198702261729

39 曹莹莹 女 320684198612318724

40 花艳 女 320683198608184886

41 李博颖 女 320683198610264121

42 卢培培 男 320682198411033770

43 朱雪沐 女 320611198211222627

46 张晔 男 32060219871128051x

48 范丽丽 女 320682198602212229

49 申阳 女 320611198310142120

50 王洪侠 女 210124198508162281

51 付文文 女 210502198412020944

52 王晓茹 女 32060219860326052x

53 袁俐 女 320602198609292020

54 马培培 女 3206231987081527508

55 孙伟 男 320611198611252614

56 葛添添 女 320602198606300021

57 杨倩 女 320602198509120547

60 彭泳 男 320683198511032018

61 顾文军 女 320683198812046026

68 管赟赟 女 320621198709072622

exam.txt

123123123

123123123

123123123

320602198704121529

320611198407212631

320623198502280013

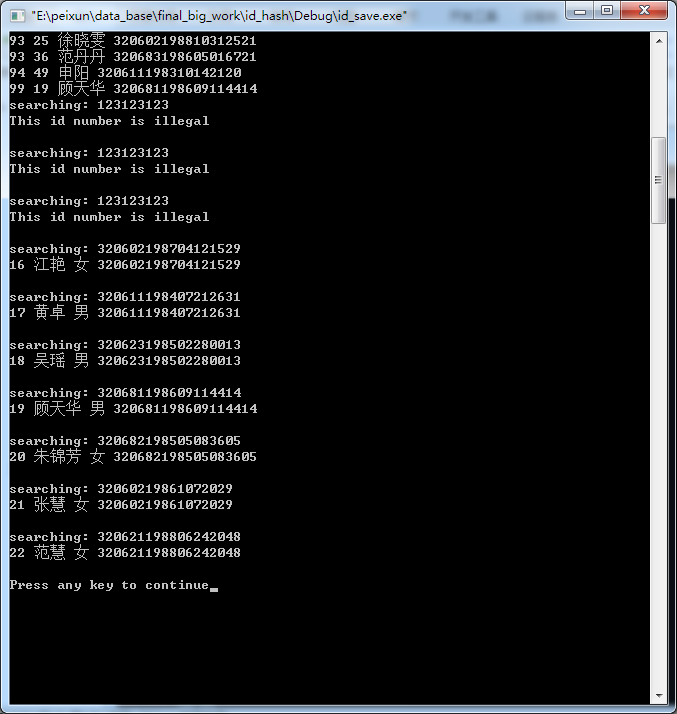
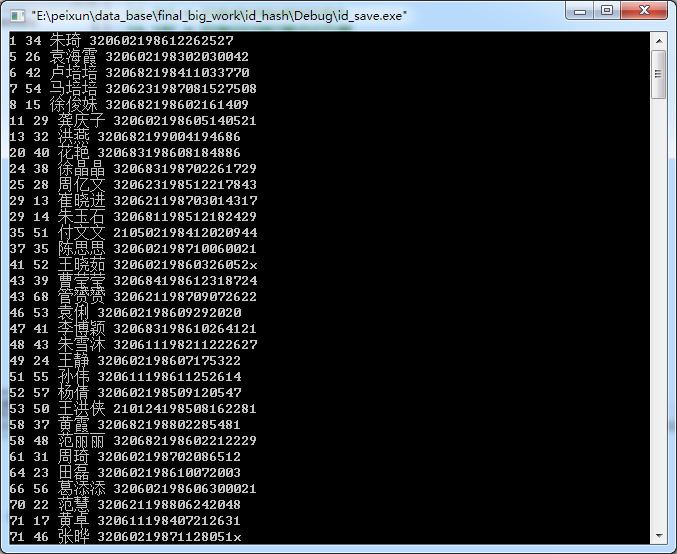
320681198609114414

320682198505083605

32060219861072029

320621198806242048

测试结果：



代码行数：

171行