

Project5_哈希表的实现_实验报告

一. 实验目的

- (1) 针对某个集体（例如是所在的班级）中的“姓名”设计一个哈希表，使得平均查找长度不超过 L ，完成相应的建表和查表程序。
 - (2) 假设姓名是汉语拼音的形式，需填入哈希表的姓名共有 30 个，取平均查找长度的上限为 2。哈希函数采用除留余数法的方式进行构造，并用伪随机探测再散列法处理冲突。
-

二.实验环境

编程语言和开发工具

编程语言：C++

开发工具：Visual Studio Code\QT

三.分析与设计

3.1 需求分析:①除留余数法的方式进行构造,并用伪随机探测再散列法处理冲突。

②额外实现链地址法处理冲突。

3.2 设计思路及细节:

1. 大概：两种冲突处理方法结果都存在一个数组里。

伪随机探测：设置 Hash 结构存储数据，通过 toascii 函数得出 key 值，将 key 值%47 得到将数据存放地址，如果数据冲突则通过 rand 函数得到伪随机值重新计算地址，直到找到数据不冲突的值。

链地址法：设置 listHash 链表结构，通过 toascii 函数得出 key 值，将 key 值%47 得到将数据存放地址，如果数据冲突则将新的数据作为旧数据的后置节点。

2. 优点：有良好 ui 界面，直观的哈希表展示。

四.代码

- 1.伪随机法以及链地址的结构实现，还有相应存储的数组

输入函数

```
struct Hash
{
    int key;
    string name;
    int st;
    Hash(){
        key=-1;
        name="";
        st=0;
    }
};

struct listHash
{
    int key;
    string name;
    int st;
    listHash *next;
    listHash(){
        key=-1;
        name="";
        st=0;
        next=NULL;
    }
};

listHash* mylHash[50];
vector<Hash> myHash(50);
vector<pair<string,int>> nametp; }

void putName(){
    for(int i=0;i<30;i++){
        string pname;
        int pkey;
        cin>>pname;
        for(auto t:pname){
            pkey+=toascii(t);
        }
        nametp.push_back({pname,pkey});
    }
}
```

- 2.伪随机探测法以及链地址法的具体实现以及相应输出。在存入数据的同时计入查找长度，如果发生冲突则查找长度++。

```

int RandMkHash(int pkey,string pnam){
    int H=pkey%47;
    if(myHash[H].key==-1){
        myHash[H].key=pkey;
        myHash[H].name=pnam;
        myHash[H].st++;
        cout<<"Final Address:"<<H<<"   Name : "<<pnam<<endl;
        return 1;
    }else{
        int cnt=0;
        for(int i=0;i<50;i++){
            int Hi=(pkey+rand()%30)%47;
            cnt++;
            if(myHash[Hi].key==-1){
                myHash[Hi].key=pkey;
                myHash[Hi].name=pnam;
                myHash[Hi].st=cnt;
                cout<<"Final Address:"<<Hi<<"   Name : "<<pnam<<endl;
                return 1;
            }else{
                cout<<"Conflict with : "<<Hi<<"   "<<myHash[Hi].name<<endl;
            }
        }
    }
    return 0;
}

```

```

int LineMkHash(int pkey,string pnam){
    int H=pkey%47;
    if(myHash[H]->key==-1){
        myHash[H]->key=pkey;
        myHash[H]->name=pnam;
        myHash[H]->st++;
        cout<<"Final Address:"<<H<<"   Name : "<<pnam<<endl;
        return 1;
    }else{
        listHash *p=myHash[H];
        while(p->next!=NULL){
            p=p->next;
        }
        listHash * temp=new listHash();
        temp->key=pkey;
        temp->name=pnam;
        temp->st=p->st++;
        p->next=temp;
        cout<<"Final Address is at: "<<H<<"   Name : "<<pnam<<" after : "<<p->name<<endl;
    }
    return 0;
}

```

3.伪随机探测法以及链地址法的查找函数，线性查找即可

```
void search(string sname){
    int nf=0;
    for(int i=0;i<myHash.size();i++,nf++){
        if(sname==myHash[i].name){
            cout<<sname<<"is at :"<<i<<endl;
            cout<<"It's search length is :"<<myHash[i].st<<endl;
            return;
        }
    }
    if(nf!=myHash.size()){
        cout<<sname<<"isn't in this table"<<endl;
        return;
    }
}
```

```
void listSearch(string sname){
    int kk=0;
    for(auto t:nametp){
        if(sname==t.first){
            kk=t.second;
        }
    }
    int H=kk%47;
    if(mylHash[H]->key==-1){
        cout<<sname<<"isn't in this table"<<endl;
        return;
    }
    if(mylHash[H]->name==sname){
        cout<<sname<<"is at :"<<H<<endl;
        cout<<"It's search length is :"<<mylHash[H]->st<<endl;
        return;
    }
    string prename=mylHash[H]->name;
    listHash *temp=mylHash[H]->next;
    while(temp!=NULL){
        if(temp->name==sname){
            cout<<sname<<"is at :"<<H<<" after :"<<prename<<endl;
            cout<<"It's search length is :"<<temp->st<<endl;
            return;
        }
        prename=temp->name;
        temp=temp->next;
    }
    return;
}
```

5.两种冲突处理方式的平均查找长度计算

```
double ASLcnt(double p){
    for(int i=0;i<50;i++){
        if(myHash[i].key!=-1){
            p+=myHash[i].st;
        }
    }
    p=p/30;
    return p;
}

double listASLcnt(double p){
    for(int i=0;i<50;i++){
        p+=mylHash[i]->st;
        while(mylHash[i]->next!=NULL){
            p+=mylHash[i]->st;
            mylHash[i]=mylHash[i]->next;
        }
    }
    p=p/30;
    return p;
}
```

6.Qt 中的建表以及查找按钮控件实现

```
void Widget::on_mkhash_clicked()
{
    QStandardItemModel *modelt = static_cast<QStandardItemModel*>(ui->hashm->model()); // 创建模型指定父类
    ui->hashm->setModel(modelt);
    for(int i=0;i<50;i++){
        modelt->setItem(i, 0, new QStandardItem(randhash[i]));
        modelt->setHeaderData(i,Qt::Vertical, i);
    }
    for(int i=0;i<randmsg->size()+1;i++){
        ui->msg->append(randmsg[i]);
    }
    ui->rasl->setText(ravl);
}

void Widget::on_search_clicked()
{
    bool isOK;
    searchone = QInputDialog::getText(NULL, "Input ", "Please input your the name to be searched", QLineEdit::Normal, "", &isOK);
    int nf=0;
    QString t="";
    for(int i=0;i<50;i++){
        if(searchone==randhash[i]){
            t+= searchone+" is at : "+QString::number(i,10,0)+"\n"+"It's search length is : "+QString::number(i,10,0)+"\n";
            nf++;
        }
    }
    if(nf==0){
        t= searchone+" isn't in this table";
    }
    if(isOK) {
        QMessageBox::information(NULL, "Result", t, QMessageBox::Yes | QMessageBox::No, QMessageBox::Yes);
    }
}
```

```

void Widget::on_lmhash_clicked()
{
    for(int i=0;i<50;i++){
        mylHash[i]=new listHash();
    }
    for(int i=0;i<30;i++){
        LineMkHash(nap[i].second,nap[i].first);
    }
    for(int i=0;i<listm.size().+1;i++){      ▲ comparison of integers of different signs: 'int' and 'unsigned lo
        ui->lmsg->append(listm[i]);
    }

    QStandardItemModel *modelt = static_cast<QStandardItemModel*>(ui->listshow->model()); // 创建模型指定父类
    ui->listshow->setModel(modelt);
    for(int i=0;i<50;i++){
        int cnt=0;
        modelt->setItem(i, cnt, new QStandardItem(QString(QString::fromLocal8Bit(mylHash[i]->name.c_str())));
        listHash* p=new listHash();
        p->next=mylHash[i];
        p=p->next;
        while(p->next!=NULL){
            p=p->next;
            cnt=cnt+1;
            modelt->setItem(i, cnt, new QStandardItem(QString(QString::fromLocal8Bit(p->name.c_str()))));
        }
        modelt->setHeaderData(i,Qt::Vertical, i);
    }
}

```

```

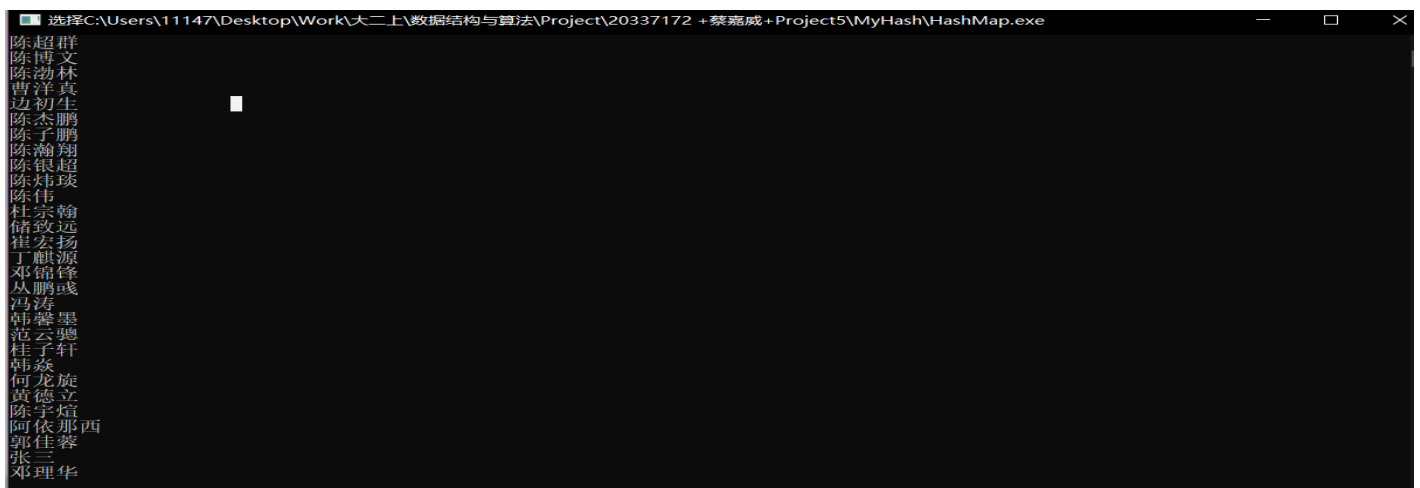
void Widget::on_lsrch_clicked()
{
    for(int i=0;i<50;i++){
        mylHash[i]=new listHash();
    }
    for(int i=0;i<30;i++){
        LineMkHash(nap[i].second,nap[i].first);
    }
    bool isOK;
    searchone = QInputDialog::getText(NULL, "Input ", "Please input your the name to be searched", QLineEdit::No
    QByteArray o=searchone.toLocal8Bit();
    string j=string(o);
    int kk=0;
    for(auto t:j){
        kk+=toascii(t);
    }
    QString yy=listSearch(kk,j);
    if(isOK) {
        QMessageBox::information(NULL, "Result",yy, QMessageBox::Yes | QMessageBox::No,QMessageBox::Yes)
    }
}

```

五. 实验结果

1. 输入

黑框输入



2.输出


伪随机： 点击 makehashmao 按钮完成图中表格建立，右边信息为建表过程各个数据的情况，包括最终地址，冲突情况，下方 ASL 处也显示平均查找长度。

Widget

RandList

MakeHashmap

Search



ASL

1. 23333

	姓名
0	储致远
1	
2	陈子鹏
3	
4	韩焱
5	阿依那西
6	丛鹏或
7	
8	曹洋真
9	郭佳蓉
10	
11	何龙旋
12	范云骢
13	陈超群
14	
15	
16	

Final Address : 40 Name : 陈子鹏

Final Address : 8 Name : 曹洋真

Final Address : 37 Name : 边初生

Final Address : 32 Name : 陈杰鹏

Final Address : 2 Name : 陈子鹏

Final Address : 21 Name : 陈瀚翔

Final Address : 38 Name : 陈银超

Final Address : 45 Name : 陈炜琰

Final Address : 19 Name : 陈伟

Conflict with : 2 陈子鹏

Final Address : 36 Name : 杜宗翰

Final Address : 0 Name : 储致远

Final Address : 42 Name : 崔宏扬

Final Address : 23 Name : 丁麒麟

Final Address : 22 Name : 邓锦锋

Final Address : 6 Name : 丛鹏或

Final Address : 34 Name : 冯涛

Final Address : 41 Name : 韩馨墨

Final Address : 12 Name : 范云骢

Conflict with : 41 韩馨墨

Conflict with : 45 陈炜琰

Conflict with : 37 边初生

Final Address : 28 Name : 桂子轩

Final Address : 4 Name : 韩焱

Final Address : 11 Name : 何龙旋

Final Address : 18 Name : 黄德立

Final Address : 31 Name : 陈宇煊

Final Address : 5 Name : 阿依那西

Final Address : 9 Name : 郭佳蓉

Conflict with : 23 丁麒麟

Conflict with : 2 陈子鹏

Conflict with : 23 丁麒麟

Final Address : 33 Name : 张三

Final Address : 26 Name : 邓理华

查找程序：当 search 按钮按下 输出为数据位置以及相应查找长度 查找失败


Input

Please input your the name to be searched

input


OKCancel

Result

 陈超群 is at : 13
It's search length is : 1

YesNo

Result

 孙二娘 isn't in this table

YesNo

链地址： 点击 makehashmao 按钮完成图中表格建立，右边信息为建表过程各个数据的情况，包括最终地址，冲突情况，下方 ASL 处也显示平均查找长度。


HashMap

Rand

List

MakeHashmap

Search



ASL

1. 3333

	姓名	2
0	储致远	阿依那西
1		
2	陈子鹏	丛鹏或
3		
4	韩焱	
5		
6		
7		
8	曹洋真	陈伟
9	郭佳蓉	
10		
11	何龙旋	
12	范云骢	
13	陈超群	
14		
15		
16		

Final Address : 44 Name : 蔡嘉威
Final Address : 13 Name : 陈超群
Final Address : 25 Name : 陈博文
Final Address : 40 Name : 陈渤林
Final Address : 8 Name : 曹洋真
Final Address : 37 Name : 边初生
Final Address : 32 Name : 陈杰鹏
Final Address : 2 Name : 陈子鹏
Final Address : 21 Name : 陈瀚翔
Final Address : 38 Name : 陈银超
Final Address : 45 Name : 陈炜琰
Final Address : 8 Name : 陈伟 after :曹洋真
Final Address : 32 Name : 杜宗翰 after :陈杰鹏
Final Address : 0 Name : 储致远
Final Address : 32 Name : 崔宏扬 after :杜宗翰
Final Address : 23 Name : 丁麒麟
Final Address : 40 Name : 邓锦锋 after :陈渤林
Final Address : 2 Name : 丛鹏或 after :陈子鹏
Final Address : 34 Name : 冯涛
Final Address : 23 Name : 韩睿墨 after :丁麒麟
Final Address : 12 Name : 范云骢
Final Address : 23 Name : 桂子轩 after :韩睿墨
Final Address : 4 Name : 韩焱
Final Address : 11 Name : 何龙旋
Final Address : 18 Name : 黄德立
Final Address : 31 Name : 陈宇煊
Final Address : 0 Name : 阿依那西 after :储致远
Final Address : 9 Name : 郭佳蓉
Final Address : 22 Name : 张三
Final Address : 26 Name : 邓理华

查找程序:当 search 按钮按下 输出为数据位置以及相应查找长度,前驱节点 查找失败

Input


Please input your the name to be searched

input

OK

Cancel

Result




阿依那西 is at : 0 after: 储致远
It's search length is : 2

Yes

No

Result



王⑤ isn't in this table

Yes

No

*因为不同软件编码形式不同，vscode 与 qt 中文编码形式不一，所导致结果也会有所差距。

Vscode 代码结果如下：

伪随机：


```
Final Address:44 Name :蔡嘉威
Final Address:10 Name :陈超群
Final Address:35 Name :陈博文
Final Address:28 Name :陈渤林
Final Address:36 Name :曹洋真
Final Address:26 Name :边初生
Final Address:11 Name :陈杰鹏
Final Address:13 Name :陈子鹏
Final Address:34 Name :陈瀚翔
Final Address:25 Name :陈银超
Final Address:23 Name :陈炜琰
Final Address:31 Name :陈伟
Final Address:16 Name :杜宗翰
Final Address:27 Name :储致远
Final Address:1 Name :崔宏扬
Final Address:24 Name :丁麒源
Final Address:17 Name :邓锦锋
Final Address:19 Name :丛鹏或
Final Address:6 Name :冯涛
Final Address:29 Name :韩馨墨
Final Address:41 Name :范云骢
Conflict with :34 陈瀚翔
Final Address:21 Name :桂子轩
Conflict with :31 陈伟
Final Address:3 Name :韩焱
Final Address:32 Name :何龙旋
Final Address:7 Name :黄德立
Final Address:5 Name :陈宇煊
Conflict with :5 陈宇煊
Final Address:9 Name :阿依那西
Final Address:43 Name :郭佳蓉
Final Address:18 Name :张三
Conflict with :11 陈杰鹏
Final Address:2 Name :邓理华 The ASL :1.13333
```

链地址：

```
Final Address:44 Name :蔡嘉威
Final Address:10 Name :陈超群
Final Address:35 Name :陈博文
Final Address:28 Name :陈渤林
Final Address:36 Name :曹洋真
Final Address:26 Name :边初生
Final Address:11 Name :陈杰鹏
Final Address:13 Name :陈子鹏
Final Address:34 Name :陈瀚翔
Final Address:25 Name :陈银超
Final Address:23 Name :陈炜琰
Final Address:31 Name :陈伟
Final Address:16 Name :杜宗翰
Final Address is at: 16 Name :储致远 after :杜宗翰
Final Address:1 Name :崔宏扬
Final Address:24 Name :丁麒源
Final Address:17 Name :邓锦锋
Final Address:19 Name :丛鹏或
Final Address:6 Name :冯涛
Final Address:29 Name :韩馨墨
Final Address:41 Name :范云骢
Final Address is at: 17 Name :桂子轩 after :邓锦锋
Final Address:21 Name :韩焱
Final Address:32 Name :何龙旋
Final Address:3 Name :黄德立
Final Address is at: 34 Name :陈宇煊 after :陈瀚翔
Final Address is at: 34 Name :阿依那西 after :陈宇煊
Final Address:43 Name :郭佳蓉
Final Address:18 Name :张三
Final Address is at: 44 Name :邓理华 after :蔡嘉威 The ASL :1.3
```

六. 压缩包文件说明

- 1.该程序先在 VScode 上完成，后转移到 QT 进行 ui 设计。
- 2.Hashmap.cpp 是基础代码文件。
3. HashMap 文件夹中为相应 QT 代码文件。
4. MyHashmap 文件夹中为完成的 exe 程序。
- 5.Test01 是测试案例。