数据结构上机实验题(三)实验报告

题目: 哈希表设计
姓名：王磊
班级：2020211311
学号：2020211538
提交日期：2021年12月12日

一 题目描述

1. 输入形式：三十个中国人姓名的汉语拼音形式构成的字符串。
2. 输出形式：每个关键字在哈希表中的位置和平均查找长度。
3. 程序功能：对读入的三十个名字使用除留余数法构造哈希函数，用伪随机探测再散列法处理冲突，伪随机序列为指定数组。

二 程序设计

1. **定义结构变量**
    1. 定义结构变量stu

```
typedef struct {
    string name;
    int len;
    int key;
}stu;
```

    2. 定义结构变量HashTable

```
typedef struct {
    stu list[HASH_LENGTH];
    int count;
}HashTable;
```

2. **基本操作**

```
void InitNameList(stu*&namelist)
    //名单初始化，存入namelist结构数组
void  CreateHash(stu*namelist,HashTable &ht)
    //为namelist创建哈希表，生成哈希表ht
int SearchHash(HashTable ht,string s,int &p,int &d)
    //在哈希表中查找字符串s，p为字符串在表中的位置，d为查找的次数
```

3. **详细设计**

1. 头文件

```
#include<cstdio>
#include<string>
#include<cstdlib>
#include<cctype>
```

2. 外部变量定义

```
const int NAME_SIZE = 20;
const int HASH_LENGTH = 53;
```

3. 结构体定义

```
typedef struct {
    string name;
    int len;
    int key;
}stu;
typedef struct {
    stu list[HASH_LENGTH];
    int count;
}HashTable;
```

4. 名单初始化

```
void InitNameList(stu*&namelist) {
    namelist = (stu*)malloc(30 * sizeof(stu));
    namelist[0].name = "zhenghao";
    namelist[0].len = size(namelist[0].name);
    namelist[1].name = "kangyuping";
    namelist[1].len = size(namelist[1].name);
    namelist[2].name = "maziyan";
    namelist[2].len = size(namelist[2].name);
    namelist[3].name = "renyiran";
    namelist[3].len = size(namelist[3].name);
    namelist[4].name = "tangziwei";
    namelist[4].len = size(namelist[4].name);
    namelist[5].name = "wangwendan";
    namelist[5].len = size(namelist[5].name);
    namelist[6].name = "zuxiyue";
    namelist[6].len = size(namelist[6].name);
    namelist[7].name = "baihaonan";
    namelist[7].len = size(namelist[7].name);
    namelist[8].name = "chenlong";
    namelist[8].len = size(namelist[8].name);
    namelist[9].name = "chengxijie";
    namelist[9].len = size(namelist[9].name);
    namelist[10].name = "guoziyi";
    namelist[10].len = size(namelist[10].name);
    namelist[11].name = "jiweijie";
    namelist[11].len = size(namelist[11].name);
    namelist[12].name = "lichangwei";
    namelist[12].len = size(namelist[12].name);
    namelist[13].name = "yixuchen";
    namelist[13].len = size(namelist[13].name);
    namelist[14].name = "liuhaisu";
    namelist[14].len = size(namelist[14].name);
    namelist[15].name = "liusijun";
    namelist[15].len = size(namelist[15].name);
    namelist[16].name = "longzhen";
    namelist[16].len = size(namelist[16].name);
    namelist[17].name = "biyang";
    namelist[17].len = size(namelist[17].name);
    namelist[18].name = "hanbinzhuo";
    namelist[18].len = size(namelist[18].name);
    namelist[19].name = "houdezheng";
    namelist[19].len = size(namelist[19].name);
    namelist[20].name = "xuhaoyang";
    namelist[20].len = size(namelist[20].name);
    namelist[21].name = "yuze";
    namelist[21].len = size(namelist[21].name);
    namelist[22].name = "yangtong";
    namelist[22].len = size(namelist[22].name);
    namelist[23].name = "jinpeng";
    namelist[23].len = size(namelist[23].name);
    namelist[24].name = "maran";
    namelist[24].len = size(namelist[24].name);
    namelist[25].name = "chenfei";
    namelist[25].len = size(namelist[25].name);
    namelist[26].name = "guokangxin";
    namelist[26].len = size(namelist[26].name);
    namelist[27].name = "hushengchun";
    namelist[27].len = size(namelist[27].name);
    namelist[28].name = "luozhenyu";
    namelist[28].len = size(namelist[28].name);
    namelist[29].name = "aiziheer";
    namelist[29].len = size(namelist[29].name);
    for (int i = 0; i < 30; i++) {
        for (int j = 0; j < namelist[i].len; j++)
            namelist[i].key += namelist[i].name[j];
    }
}
```

5. 创建哈希表

```
void  CreateHash(stu*namelist,HashTable &ht) {
    for (int i = 0; i < 53; i++) {
        ht.list[i].name = "\0";
        ht.list[i].len = 0;
        ht.list[i].key = 0;
    }
    ht.count = 0;
    int hash;
    int d = 0;

    for (int i = 0; i < 30; i++) {
        int rand[53] = { 98,82 , 29 , 49,  45  ,17 , 64,  4,  23  ,79  ,93  ,44 , 32  ,17  ,21  ,26,  53
, 65  ,31,  29  ,70 , 89 , 12 , 4  ,77 , 85  ,7  ,77 , 9 , 97 , 61 , 75,  62,  62  ,54  ,71 , 6 , 5  ,98
,39  ,53 , 20 , 48,  13 , 74,  72,  4  ,59 , 61 , 98  ,76  ,26 , 68 };
        d = 0;
        hash = namelist[i].key % 53 + d;
        while (ht.list[hash].name != "\0") {
            hash = (hash + rand[d]) % 53;
            d++;
        }
        ht.list[hash] = namelist[i];
```

```
            ht.count++;
        }
    }
```

6. 查找哈希表

```
int SearchHash(HashTable ht,string s,int &p,int &d) {
    int rand[53] = { 98,82 , 29 , 49,  45  ,17 , 64,  4,  23  ,79  ,93  ,44 , 32  ,17  ,21  ,26,  53 , 65 ,31,  29  ,70 , 89 , 12 , 4  ,77 , 85  ,7  ,77 , 9 , 97 , 61 , 75,  62,  62  ,54  ,71 , 6 , 5  ,98  ,39 ,53 , 20 , 48,  13 , 74,  72,  4  ,59 , 61 , 98  ,76  ,26 , 68 };
    int i,key,flag,hash;
    i = 0;
    d = 0;
    key = 0;
    flag = 0;
    while (s[i]) {
        key += s[i];
        i++;
    }
    hash = key % 53 + d;
    while (ht.list[hash].name != "\0") {
        if (ht.list[hash].key == key) {
            p = hash;
            flag = 1;
        }
        hash = (hash + rand[d]) % 53;
        d++;
    }
    return flag;
}
```

7. 主函数

```
int main()
{
    stu* namelist;
    HashTable ht;
    int p, d, sum;
    p = 0;
    sum = 0;
    InitNameList(namelist);
    CreateHash(namelist, ht);
    for (int i = 0; i < 30; i++) {
        SearchHash(ht, namelist[i].name, p, d);
        printf("%d\n", p);
        sum += d;
    }
    printf("Average search times:%d\n", sum / 30 + 1);
}
```

三 源代码

```
#include<cstdio>
#include<string>
#include<cstdlib>
#include<cctype>
using namespace std;
const int NAME_SIZE = 20;
const int HASH_LENGTH = 53;
typedef struct {
    string name;
    int len;
    int key;
}stu;
typedef struct {
    stu list[HASH_LENGTH];
    int count;
}HashTable;
void InitNameList(stu*&namelist) {
    namelist = (stu*)malloc(30 * sizeof(stu));
    namelist[0].name = "zhenghao";
    namelist[0].len = size(namelist[0].name);
    namelist[1].name = "kangyuping";
    namelist[1].len = size(namelist[1].name);
    namelist[2].name = "maziyan";
    namelist[2].len = size(namelist[2].name);
    namelist[3].name = "renyiran";
    namelist[3].len = size(namelist[3].name);
    namelist[4].name = "tangziwei";
    namelist[4].len = size(namelist[4].name);
    namelist[5].name = "wangwendan";
    namelist[5].len = size(namelist[5].name);
    namelist[6].name = "zuxiyue";
    namelist[6].len = size(namelist[6].name);
    namelist[7].name = "baihaonan";
    namelist[7].len = size(namelist[7].name);
    namelist[8].name = "chenlong";
    namelist[8].len = size(namelist[8].name);
    namelist[9].name = "chengxijie";
    namelist[9].len = size(namelist[9].name);
    namelist[10].name = "guoziyi";
    namelist[10].len = size(namelist[10].name);
    namelist[11].name = "jiweijie";
    namelist[11].len = size(namelist[11].name);
    namelist[12].name = "lichangwei";
    namelist[12].len = size(namelist[12].name);
    namelist[13].name = "yixuchen";
    namelist[13].len = size(namelist[13].name);
    namelist[14].name = "liuhaisu";
    namelist[14].len = size(namelist[14].name);
    namelist[15].name = "liusijun";
    namelist[15].len = size(namelist[15].name);
    namelist[16].name = "longzhen";
    namelist[16].len = size(namelist[16].name);
    namelist[17].name = "biyang";
    namelist[17].len = size(namelist[17].name);
    namelist[18].name = "hanbinzhuo";
    namelist[18].len = size(namelist[18].name);
    namelist[19].name = "houdezheng";
    namelist[19].len = size(namelist[19].name);
    namelist[20].name = "xuhaoyang";
    namelist[20].len = size(namelist[20].name);
    namelist[21].name = "yuze";
    namelist[21].len = size(namelist[21].name);
    namelist[22].name = "yangtong";
    namelist[22].len = size(namelist[22].name);
    namelist[23].name = "jinpeng";
    namelist[23].len = size(namelist[23].name);
    namelist[24].name = "maran";
    namelist[24].len = size(namelist[24].name);
    namelist[25].name = "chenfei";
    namelist[25].len = size(namelist[25].name);
    namelist[26].name = "guokangxin";
    namelist[26].len = size(namelist[26].name);
    namelist[27].name = "hushengchun";
    namelist[27].len = size(namelist[27].name);
    namelist[28].name = "luozhenyu";
    namelist[28].len = size(namelist[28].name);
    namelist[29].name = "aiziheer";
    namelist[29].len = size(namelist[29].name);
    for (int i = 0; i < 30; i++) {
        for (int j = 0; j < namelist[i].len; j++)
            namelist[i].key += namelist[i].name[j];
    }
}
void  CreateHash(stu*namelist,HashTable &ht) {
    for (int i = 0; i < 53; i++) {
        ht.list[i].name = "\0";
        ht.list[i].len = 0;
        ht.list[i].key = 0;
    }
    ht.count = 0;
    int hash;
    int d = 0;

    for (int i = 0; i < 30; i++) {
        int rand[53] = { 98,82 , 29 , 49,  45  ,17 , 64,  4,  23  ,79  ,93  ,44 , 32  ,17  ,21  ,26,  53
, 65  ,31,  29  ,70 , 89 , 12 , 4  ,77 , 85  ,7  ,77 , 9 , 97 , 61 , 75,  62,  62  ,54  ,71 , 6 , 5  ,98
,39  ,53 , 20 , 48,  13 , 74,  72,  4  ,59 , 61 , 98  ,76  ,26 , 68 };
        d = 0;
        hash = namelist[i].key % 53 + d;
        while (ht.list[hash].name != "\0") {
            hash = (hash + rand[d]) % 53;
            d++;
        }
        ht.list[hash] = namelist[i];
        ht.count++;
    }
}
int SearchHash(HashTable ht,string s,int &p,int &d) {
    int rand[53] = { 98,82 , 29 , 49,  45  ,17 , 64,  4,  23  ,79  ,93  ,44 , 32  ,17  ,21  ,26,  53 , 65
,31,  29  ,70 , 89 , 12 , 4  ,77 , 85  ,7  ,77 , 9 , 97 , 61 , 75,  62,  62  ,54  ,71 , 6 , 5  ,98  ,39
,53 , 20 , 48,  13 , 74,  72,  4  ,59 , 61 , 98  ,76  ,26 , 68 };
    int i,key,flag,hash;
    i = 0;
    d = 0;
    key = 0;
    flag = 0;
    while (s[i]) {
        key += s[i];
        i++;
    }
    hash = key % 53 + d;
    while (ht.list[hash].name != "\0") {
        if (ht.list[hash].key == key) {
            p = hash;
            flag = 1;
        }
        hash = (hash + rand[d]) % 53;
        d++;
    }
    return flag;
}
int main()
{
    stu* namelist;
    HashTable ht;
    int p, d, sum;
    p = 0;
    sum = 0;
    InitNameList(namelist);
    CreateHash(namelist, ht);
    for (int i = 0; i < 30; i++) {
        SearchHash(ht, namelist[i].name, p, d);
        printf("%d\n", p);
        sum += d;
    }
    printf("Average search times:%d\n", sum / 30 + 1);
}
```