**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**

1、购物网站信息管理（必做）（链表）

[问题描述]

设计一个程序，对商铺信息管理，商铺信息包括：商铺编号，商铺名，信誉度，（商品名称1，价格1，销量1），（商品名称2，价格2，销量2），（商品名称3，价格3，销量3）…。

假设商品名称包括(衣服，裤子，鞋子，手套，袜子，帽子等)，每个商铺具有其中若干商品。价格、销量随机确定。

[基本要求]

（1）建立一个单向链表存储所有商铺信息（至少30个），以编号为序，编号从1开始递增，从文件中读取数据，并能将数据存储在文件。商铺信息结点的数据结构自行设计。

（2）可以增、删商铺。增加商铺，编号自动加一，插入链表尾部；删除商铺以编号为准，并修改后续结点的编号，保持编号连续性。

（3）可以增、删选定商铺中的商品，修改商品价格。

（4）查询某一种商品名称，建立一个双向循环链表，结点信息是包含该商品的所有商铺编号，商铺名，信誉度，商品名称，价格，销量，按销量高至低排序。

（5）购买某一商铺的商品，修改单向链表中商品的信息的销量，修改双向循环链表中销量高至低排序。

（6）任何的信息变化，可以实现文件存储。

采用的数据结构：

链表

算法设计思想：

1. 通过链表存储商店信息
2. 通过链表存储商品信息
3. 在商店信息的结构体中新建商品指针指向商品
4. 将每次数据改动都存到文件中 每次操作都有对应的 文件存储

源代码：

#include<iostream>

#include<map>

#include<string>

#include<string.h>

#include<fstream>

using namespace std;

typedef struct shop\_info

{

int id;

string name;

int credity;

int product\_num;

shop\_info(int a,string b,int c,int d)

{

id=a;

name=b;

credity=c;

product\_num=d;

}

shop\_info()

{

id=0;

name=' ';

credity=0;

product\_num=0;

}

}shop\_info;

typedef struct product

{

int price;

int sale\_num;

string product\_name;

product \*next\_same\_product,\*last\_same\_product;

product \*shop\_next\_product;

shop\_info \*belong\_shop;

product(int a,int b,shop\_info \*c,string d)

{

price=a;

sale\_num=b;

next\_same\_product=NULL;

last\_same\_product=NULL;

shop\_next\_product=NULL;

belong\_shop=c;

product\_name=d;

}

product()

{

price=0;

sale\_num=0;

next\_same\_product=NULL;

last\_same\_product=NULL;

shop\_next\_product=NULL;

belong\_shop=NULL;

product\_name=' ';

}

}product;

typedef struct shop

{

shop\_info shop\_main;

product \*first\_product;

shop \*next\_shop;

shop()

{

first\_product=new product;

next\_shop=NULL;

}

}shop;

typedef struct all\_shop

{

shop root;

shop \*head,\*tail;

int shop\_num;

int product\_num;

map<string,int>product\_match;

product all\_pro[100];

all\_shop()

{

head=&root;

tail=NULL;

shop\_num=0;

product\_num=0;

}

}all\_shop;

void insert\_product(product all\_pro[],product \*pro,int id)

{

product \*p;

if(all\_pro[id].next\_same\_product==NULL)

{

all\_pro[id].next\_same\_product=pro;

}

else

{

p=&all\_pro[id];

while(p->next\_same\_product&&(p->next\_same\_product->sale\_num>pro->sale\_num))

{

p=p->next\_same\_product;

}

pro->last\_same\_product=p;

pro->next\_same\_product=p->next\_same\_product;

if(p->next\_same\_product)

{

p->next\_same\_product->last\_same\_product=pro;

}

p->next\_same\_product=pro;

}

//cout<<"ook"<<endl;

}

void display\_product(all\_shop All\_s)

{

product \*p;

for(int i=0;i<All\_s.product\_num;i++)

{

//int i=1;

p=All\_s.all\_pro[i].next\_same\_product;

while(p!=NULL)

{

cout<<p->belong\_shop->name<<" "<<p->product\_name<<" "<<p->price<<" "<<p->sale\_num<<endl;

p=p->next\_same\_product;

}

}

}

void display\_shops(all\_shop All\_s)

{

shop \*p,\*head;

product \*q;

head=&All\_s.root;

p=head->next\_shop;

while(p)

{

cout<<p->shop\_main.id<<" "<<p->shop\_main.name<<" "<<p->shop\_main.credity<<" "<<p->shop\_main.product\_num<<endl;

q=p->first\_product->shop\_next\_product;

while(q)

{

cout<<q->product\_name<<" "<<q->price<<" "<<q->sale\_num<<endl;

q=q->shop\_next\_product;

}

p=p->next\_shop;

}

}

void write\_shop(shop head,char \*save\_file)

{

fstream f1;

f1.open(save\_file,ios::out);

shop \*p;

product \*q;

p=head.next\_shop;

while(p)

{

f1<<p->shop\_main.id<<" "<<p->shop\_main.name<<" "<<p->shop\_main.credity<<" "<<p->shop\_main.product\_num<<" ";

q=p->first\_product->shop\_next\_product;

while(q)

{

f1<<q->product\_name<<" "<<q->price<<" "<<q->sale\_num<<" ";

q=q->shop\_next\_product;

}

f1<<"\n";

p=p->next\_shop;

}

f1.close();

}

void read\_shop(all\_shop &All\_s)

{

fstream f1;

shop \*s,\*p;

string pro\_name;

product \*pro;

// product all\_pro[100];

product \*all\_pro;

all\_pro=All\_s.all\_pro;

memset(all\_pro,NULL,sizeof(all\_pro));

// map<string,int>All\_s.product\_match;

s=&All\_s.root;

//s=new shop;

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

while(!f1.eof())

{

p = new shop;

int price;

f1>>p->shop\_main.id>>p->shop\_main.name>>p->shop\_main.credity>>p->shop\_main.product\_num;

// cout<<p->shop\_main.id<<" "<<p->shop\_main.name<<" "<<p->shop\_main.credity<<" "<<p->shop\_main.product\_num<<endl;

for(int i=0;i<p->shop\_main.product\_num;i++)

{

pro = new product;

f1>>pro\_name;

if(!All\_s.product\_match.count(pro\_name))

{

All\_s.product\_match[pro\_name]=All\_s.product\_num++;

}

f1>>pro->price;

f1>>pro->sale\_num;

pro->product\_name=pro\_name;

//cout<<pro\_name<<" "<<pro->price<<" "<<pro->sale\_num<<endl;

pro->belong\_shop=&(p->shop\_main);

if(!p->first\_product->shop\_next\_product)

{

p->first\_product->shop\_next\_product=pro;

}

else

{

pro->shop\_next\_product=p->first\_product->shop\_next\_product;

p->first\_product->shop\_next\_product=pro;

}

insert\_product(all\_pro,pro,All\_s.product\_match[pro\_name]);

}

s->next\_shop=p;

s=p;

All\_s.shop\_num++;

}

All\_s.tail=p;

f1.close();

}

shop \*locate\_shop(all\_shop All\_s,string s\_name)

{

shop \*p;

p=All\_s.head->next\_shop;

while(p)

{

if(!p->shop\_main.name.compare(s\_name))

return p;

p=p->next\_shop;

}

return NULL;

}

shop \*locate\_shop\_p(all\_shop All\_s,string s\_name)//locate\_parent

{

shop \*p;

p=All\_s.head;

while(p->next\_shop)

{

if(!p->next\_shop->shop\_main.name.compare(s\_name))

return p;

p=p->next\_shop;

}

return NULL;

}

product\* locate\_product(shop \*p,string pro\_name)

{

product \*q;

q=p->first\_product;

while(q)

{

if(q->product\_name.compare(pro\_name)==0)

return q;

q=q->shop\_next\_product;

}

return NULL;

}

product\* locate\_product\_p(shop \*p,string pro\_name)

{

product \*q;

q=p->first\_product;

while(q->shop\_next\_product)

{

if(q->shop\_next\_product->product\_name.compare(pro\_name)==0)

return q;

q=q->shop\_next\_product;

}

return NULL;

}

void adjust\_product(product \*pro)

{

product \*p;

p=pro;

if(p)

cout<<p->product\_name<<p->sale\_num<<endl;

if(p->last\_same\_product||p->next\_same\_product)

{

if(p->last\_same\_product->last\_same\_product&&(p->last\_same\_product->sale\_num<pro->sale\_num))

{

p=p->last\_same\_product;

while(p->last\_same\_product&&(p->sale\_num<pro->sale\_num))

p=p->last\_same\_product;

cout<<"locate:"<<p->product\_name<<p->sale\_num<<endl;

}

else if(p->next\_same\_product&&(p->next\_same\_product->sale\_num>pro->sale\_num))

{

cout<<"locate:"<<p->product\_name<<p->sale\_num<<endl;

p=p->next\_same\_product;

while(p->next\_same\_product&&(p->sale\_num>pro->sale\_num))

{

p=p->next\_same\_product;

}

cout<<"locate:"<<p->product\_name<<p->sale\_num<<endl;

if(p->next\_same\_product)

{

p=p->last\_same\_product;

}

cout<<"locate:"<<p->product\_name<<p->sale\_num<<endl;

}

if(p!=pro)

{

if(pro->last\_same\_product)

pro->last\_same\_product->next\_same\_product=pro->next\_same\_product;

if(pro->next\_same\_product)

pro->next\_same\_product->last\_same\_product=pro->last\_same\_product;

pro->last\_same\_product=p;

pro->next\_same\_product=p->next\_same\_product;

if(p->next\_same\_product)

p->next\_same\_product->last\_same\_product=pro;

p->next\_same\_product=pro;

}

}

}

void add\_product(all\_shop &All\_s)

{

//locate the shop

fstream f1;

int n;

f1.open("add\_pro.txt",ios::in);

while(!f1.eof())

{

string add\_s;

shop \*p;

int i=0;

product \*pro;

f1>>add\_s;

f1>>n;

p=locate\_shop(All\_s,add\_s);

if(p)

{

cout<<"found"<<endl;

cout<<p->shop\_main.product\_num<<endl;

string add\_pro;

while(!f1.eof()&&i<n)

{

f1>>add\_pro;

// cout<<add\_pro<<endl;

pro=locate\_product(p,add\_pro);

if(!pro)

{

// cout<<"null"<<add\_pro<<endl;

// cout<<add\_pro<<endl;

if(!All\_s.product\_match.count(add\_pro))

{

All\_s.product\_match[add\_pro]=All\_s.product\_num++;

}

pro=new product;

pro->product\_name=add\_pro;

f1>>pro->price>>pro->sale\_num;

// cout<<"price: "<<pro->price<<"vsale: "<<pro->sale\_num<<endl;

pro->belong\_shop=&p->shop\_main;

pro->shop\_next\_product=p->first\_product->shop\_next\_product;

p->first\_product->shop\_next\_product=pro;

p->shop\_main.product\_num++;

insert\_product(All\_s.all\_pro,pro,All\_s.product\_match[add\_pro]);

}

else

{

f1>>pro->price>>pro->sale\_num;

pro->product\_name=add\_pro;

adjust\_product(pro);

}

i++;

}

}

}

f1.close();

}

void add\_shop(all\_shop &All\_s)

{

shop \*p;

fstream f1;

shop \*s;

string pro\_name;

product \*pro;

product \*all\_pro;

all\_pro=All\_s.all\_pro;

memset(all\_pro,NULL,sizeof(all\_pro));

s=All\_s.tail;

f1.open("new\_shop.txt",ios::in);

while(!f1.eof())

{

p=new shop;

f1>>p->shop\_main.name>>p->shop\_main.credity>>p->shop\_main.product\_num;

p->shop\_main.id=All\_s.shop\_num+1;

cout<<p->shop\_main.id<<" "<<p->shop\_main.name<<" "<<p->shop\_main.credity<<" "<<p->shop\_main.product\_num<<endl;

for(int i=0;i<p->shop\_main.product\_num;i++)

{

pro = new product;

f1>>pro\_name;

if(!All\_s.product\_match.count(pro\_name))

{

All\_s.product\_match[pro\_name]=All\_s.product\_num++;

}

f1>>pro->price;

f1>>pro->sale\_num;

pro->product\_name=pro\_name;

cout<<pro\_name<<" "<<pro->price<<" "<<pro->sale\_num<<endl;

pro->belong\_shop=&(p->shop\_main);

if(!p->first\_product->shop\_next\_product)

{

p->first\_product->shop\_next\_product=pro;

}

else

{

pro->shop\_next\_product=p->first\_product->shop\_next\_product;

p->first\_product->shop\_next\_product=pro;

}

insert\_product(all\_pro,pro,All\_s.product\_match[pro\_name]);

}

s->next\_shop=p;

s=p;

All\_s.shop\_num++;

// cout<<"ok"<<endl;

}

f1.close();

}

void buy\_product(all\_shop &All\_s)

{

fstream f1;

shop \*s\_p;

product \*pro;

f1.open("buy\_product.txt",ios::in);

string shop\_name;

string pro\_name;

int buy\_num=0;

while(!f1.eof())

{

f1>>shop\_name;

s\_p=locate\_shop(All\_s,shop\_name);

if(s\_p)

{

f1>>pro\_name;

pro=locate\_product(s\_p,pro\_name);

if(pro)

{

f1>>buy\_num;

pro->sale\_num+=buy\_num;

adjust\_product(pro);

}

else

{

cout<<"this doesn't have such product"<<endl;

}

}

else

{

cout<<"no such shop: "<<shop\_name<<endl;

}

}

}

void delete\_product(all\_shop &All\_s,product \*pro)

{

if(!pro->last\_same\_product&&!pro->next\_same\_product)

{

All\_s.all\_pro[All\_s.product\_match[pro->product\_name]].next\_same\_product=NULL;

}

else

{

if(pro->last\_same\_product)

pro->last\_same\_product->next\_same\_product=pro->next\_same\_product;

if(pro->next\_same\_product)

pro->next\_same\_product->last\_same\_product=pro->last\_same\_product;

}

free(pro);

}

void delete\_products(all\_shop &All\_s)

{

fstream f1;

string shop\_name;

string product\_name;

shop \*s\_p,\*s\_s\_p;

product \*pro,\*pro\_p;

f1.open("delete\_products.txt",ios::in);

while(!f1.eof())

{

f1>>shop\_name;

s\_p=locate\_shop(All\_s,shop\_name);

if(s\_p)

{

f1>>product\_name;

pro=locate\_product(s\_p,product\_name);

if(pro)

{

pro\_p=locate\_product\_p(s\_p,product\_name);

pro\_p->shop\_next\_product=pro->shop\_next\_product;

delete\_product(All\_s,pro);

}

else

{

cout<<endl<<"There is no "<<product\_name<<"at shop "<<shop\_name<<endl;

}

}

else

{

cout<<"no such shop"<<endl;

}

}

f1.close();

}

void delete\_shop(all\_shop &All\_s,shop \*s\_p,shop \*s\_p\_p)

{

All\_s.shop\_num--;

product \*p,\*q;

int id;

p=s\_p->first\_product;

s\_p->first\_product=NULL;

s\_p\_p->next\_shop=s\_p->next\_shop;

id=s\_p\_p->shop\_main.id+1;

s\_p\_p=s\_p\_p->next\_shop;

while(s\_p\_p)

{

s\_p\_p->shop\_main.id=id++;

s\_p\_p=s\_p\_p->next\_shop;

}

while(p)

{

q=p->shop\_next\_product;

delete\_product(All\_s,p);

p=q;

}

free(s\_p);

}

void delete\_shops(all\_shop &All\_s)

{

fstream f1;

string shop\_name;

shop \*s\_p,\*s\_p\_p;

f1.open("delete\_shop.txt",ios::in);

while(!f1.eof())

{

f1>>shop\_name;

s\_p=locate\_shop(All\_s,shop\_name);

if(s\_p)

{

s\_p\_p=locate\_shop\_p(All\_s,shop\_name);

delete\_shop(All\_s,s\_p,s\_p\_p);

}

else

{

cout<<"no such shop!"<<endl;

}

}

f1.close();

}

void search\_product(all\_shop All\_s)

{

fstream f1;

string product\_name;

product \*p;

int flag=1;

f1.open("search\_product.txt",ios::in);

while(!f1.eof())

{

f1>>product\_name;

flag=0;

cout<<"searching "<<product\_name<<endl;

if(All\_s.product\_match.count(product\_name))

{

p=All\_s.all\_pro[All\_s.product\_match[product\_name]].next\_same\_product;

while(p)

{

cout<<p->belong\_shop->name<<" "<<p->product\_name<<" "<<p->price<<" "<<p->sale\_num<<endl;

p=p->next\_same\_product;

}

flag=1;

}

if(flag==0)

cout<<"sorry can't find "<<product\_name<<endl;

}

}

int main()

{

int flag=1;

shop root;

all\_shop All\_s;

fstream f1;

//read shop infomation

product all\_pro[100];

cout<<"read shop info..."<<endl;

system("pause");

read\_shop(All\_s);

while(flag)

{

cout<<"what do you want to do next: "<<endl;

cout<<"1 : add shops"<<endl;

cout<<"2 : add products"<<endl;

cout<<"3 : display product information"<<endl;

cout<<"4 : display shop information"<<endl;

cout<<"5 : buy product"<<endl;

cout<<"6 : delete shop"<<endl;

cout<<"7 : delete shop"<<endl;

cout<<"8 : search product"<<endl;

cout<<"0 : end "<<endl;

cin>>flag;

switch(flag)

{

case 1:

cout<<"add shop info..."<<endl;

cout<<"Please make sure that you have put the shop info you want to add at 'new\_shop.txt'"<<endl;

system("pause");

add\_shop(All\_s);

cout<<"put added info into file..."<<endl;

system("pause");

write\_shop(All\_s.root,"added\_shop.txt");

cout<<"You can open added\_shop.txt to check if shops was added"<<endl;

system("pause");

break;

case 2:

cout<<"add product"<<endl;

cout<<"Please make sure that you have put the product info you want to add at 'new\_pro.txt'"<<endl;

add\_product(All\_s);

cout<<"put added product info into file.."<<endl;

system("pause");

write\_shop(All\_s.root,"added\_product.txt");

cout<<"You can open added\_shop.txt to check if products was added"<<endl;

system("pause");

break;

case 3:

cout<<"product info"<<endl;

system("pause");

display\_product(All\_s);

break;

case 4:

cout<<"shop info"<<endl;

system("pause");

display\_shops(All\_s);

break;

case 5:

cout<<"buy product.."<<endl;

cout<<"Please make sure that you have put the product info you want to buy at 'buy\_product.txt'"<<endl;

system("pause");

buy\_product(All\_s);

cout<<"update file"<<endl;

write\_shop(All\_s.root,"bought\_product.txt");

cout<<"You can open bought\_product.txt to check if products was bought"<<endl;

system("pause");

break;

case 6:

cout<<"delete shop.."<<endl;

cout<<"Please make sure that you have put the shop info you want to delete at 'delete\_shops.txt'"<<endl;

system("pause");

delete\_shops(All\_s);

cout<<"update file"<<endl;

write\_shop(All\_s.root,"deleted\_shop.txt");

cout<<"You can open deleted\_shop.txt to check if shops was deleted"<<endl;

system("pause");

break;

case 7:

cout<<endl<<"deletd product"<<endl;

cout<<"Please make sure that you have put the product info you want to delete at 'delete\_products.txt'"<<endl;

system("pause");

delete\_products(All\_s);

cout<<"update file"<<endl;

write\_shop(All\_s.root,"deleted\_product.txt");

cout<<"You can open deleted\_product.txt to check if products was deleted"<<endl;

system("pause");

break;

case 8:

cout<<"search product .."<<endl;

cout<<"Please make sure that you have put the product info you want to search at 'search\_product.txt'"<<endl;

system("pause");

search\_product(All\_s);

write\_shop(All\_s.root,"shop2.txt");

break;

default:

break;

}

}

return 0;

}

测试数据：

new\_shop.txt

new1小花 22 4 手套1 10 2 围巾 11 21 袜子 90 910 眼镜 122 930

new2小花 221 4 手套2 10 2 围巾 11 221 袜子 90 920 眼镜 122 950

new3小花 222 4 手套3 10 2 围巾 11 231 袜子 90 390 眼镜 122 960

new4小花 223 4 手套4 10 2 围巾 11 241 袜子 90 940 眼镜 122 970

add\_pro.txt

甲鱼 5

玫瑰花 12 23

茉莉花 11 88

小花 11 99

鱼 12 86

围巾 220 200

buy\_product.txt

甲鱼 围巾 3

梦瑶 手套 5

哈哈 123 123

小鹿 袜子 8

delete\_shop.txt

甲鱼 小璐 佳钰

Search\_product.txt

围巾 帽子 袜子 口罩

结果：生成文件

added\_shop.txt

added\_product.txt

deleted\_shop.txt

deleted\_product.txt

bought\_product.txt

算法复杂度：

O(n)

程序代码行：683行