// main.cpp

#include <iostream>

#include <fstream>

#include <iomanip>

#include <vector>

#include "Global.hpp"

#include "Flower.hpp"

#include "Hash.hpp"

#include "Menu.hpp"

#include "Map.hpp"

using namespace std;

HashTable hashtable;

char leftad1[30], leftad2[30], leftad3[30], leftad4[30], leftad5[30], leftad6[30], leftad7[30], leftad8[30], leftad9[30], leftad10[30];

char rightad1[30], rightad2[30], rightad3[30], rightad4[30], rightad5[30], rightad6[30], rightad7[30], rightad8[30], rightad9[30], rightad10[30];

char announcement1[100], announcement2[100], announcement3[100], announcement4[100], announcement5[100];

int main() {

hashtable.InitHashTable();

MainMenu M;

M.LoadAd();

M.LoadAnnouncement();

M.MenuRun();

return 0;

}

// Map.cpp

#include "Map.hpp"

//默认构造函数

Map::Map() {

num = 0;

for (int i = 0; i < ADDRESS\_MAX\_NUM; i++) {

merchant[i].x = 0;

merchant[i].y = 0;

merchant[i].shop\_name = "";

}

}

//读取文件建立地图

void Map::InitMap() {

ifstream OutMyFile1;

OutMyFile1.open(MAP\_FILE\_NAME);

int i = 0, j = 0;

while (!OutMyFile1.eof()) {

for (; j < MAP\_LENGTH; j++) {

OutMyFile1>>market\_map[i][j];

if (market\_map[i][j] == '@')

num ++;

}

i++;

j = 0;

}

OutMyFile1.close();

ifstream OutMyFile2;

OutMyFile2.open(MERCHANT\_ADDRESS\_FILE\_NAME);

i = 0;

while (!OutMyFile2.eof()) {

OutMyFile2>>merchant[i].shop\_name>>merchant[i].x>>merchant[i].y;

i++;

}

OutMyFile2.close();

}

//写入文件保存地图

void Map::SaveMap() {

ofstream InMyFile1;

InMyFile1.open(MAP\_FILE\_NAME);

for (int i = 0; i < MAP\_WIDTH; i++){

for (int j = 0; j < MAP\_LENGTH; j++){

InMyFile1<<market\_map[i][j];

if ( j != MAP\_LENGTH - 1)

InMyFile1 << " ";

}

if (i != MAP\_WIDTH - 1)

InMyFile1 << endl;

}

InMyFile1.close();

ofstream InMyFile2;

InMyFile2.open(MERCHANT\_ADDRESS\_FILE\_NAME);

for (int i = 0; i < num; i++) {

InMyFile2<<merchant[i].shop\_name<<" "<<merchant[i].x<<" "<<merchant[i].y;

if (i != num - 1)

InMyFile2<<endl;

}

InMyFile2.close();

}

//输出地图

void Map::ShowMap() {

const string kg = " ";

char cc = 'a';

cout << kg;

for (int i = 0; i < 4; i++)

cout << " ";

for (int i = 0; i < MAP\_LENGTH - 3; i++)

cout << char(cc + i) << " ";

cout << endl;

for (int i = 0, k = 0; i < MAP\_WIDTH; i++) {

cout << kg;

if (i > 0 && i < MAP\_WIDTH-1){

cout << char(cc + k) << " ";

k++;

}

else

cout << " ";

for (int j = 0; j < MAP\_LENGTH; j++) {

if (market\_map[i][j] == '1')

cout << "\* ";

else if (market\_map[i][j] == '#')

cout << " ";

else if (market\_map[i][j] == '+')

cout << "+ ";

else if (market\_map[i][j] == '@')

cout << "@ ";

else

cout << " " ;

}

cout << endl;

}

}

//商家入驻时修改地图

bool Map::AddAddress(string &shop\_name) {

if (num == ADDRESS\_MAX\_NUM)

return false;

for (int i = 2; i <= 18; i += 4)

for(int j = 2; j <= 26; j += 4){

if (market\_map[i][j] == '#') {

market\_map[i][j] = '@';

merchant[num].shop\_name = shop\_name;

merchant[num].x = i;

merchant[num].y = j;

num++;

return true;

}

}

return false;

}

//根据商家店名搜索商家，返回该商家所在的坐标

MapNode Map::SearchMap(string &shop\_name) {

for (int i = 0; i < num; i++) {

if (merchant[i].shop\_name == shop\_name)

return merchant[i];

}

MapNode node;

node.x = -1;

return node;

}

//根据地图坐标返回在数组中的位置

path\_node Map::locate(char x, char y) {

path\_node node;

node.x = (x - 97) + 1;

node.y = (y - 97) + 1;

return node;

}

//广度优先搜索与剪枝建立画出最短路径

void Map::ChangeMap(const MapNode &merchant\_address, path\_node &start) {

int visit[MAP\_WIDTH][MAP\_LENGTH];

for (int i = 0 ; i < MAP\_WIDTH; i++) {

memset(visit[i], 0, sizeof(int)\*MAP\_LENGTH);

}

path\_node \*q = &start;

queue<path\_node\*> bfsnode;

start.front\_node = NULL;

bfsnode.push(q);

visit[q->x][q->y] = 1;

int x\_difference = abs(start.x - merchant\_address.x); //用于剪枝的状态点

int y\_difference = abs(start.y - merchant\_address.y);

stack<path\_node\*> path;

stack<path\_node\*> memory\_management;

// int n = 0; //测试搜索次数

bool flag = false; //防止第一个点（start）入栈，因为start是函数形参,不能被释放内存

while (!bfsnode.empty()) {

path\_node \*node = bfsnode.front();

// n++; //搜索次数

bfsnode.pop();

int x\_df = abs(node->x - merchant\_address.x);

int y\_df = abs(node->y - merchant\_address.y);

// 更新状态点

if (x\_df\*x\_df + y\_df\*y\_df < x\_difference\*x\_difference + y\_difference\*y\_difference) {

x\_difference = x\_df;

y\_difference =y\_df;

}

if (node->x == merchant\_address.x && node->y == merchant\_address.y) {

while(node->front\_node != NULL) {

path.push(node);

node = node->front\_node;

}

//释放内存1

while(!bfsnode.empty()) {

path\_node \*node = bfsnode.front();

bfsnode.pop();

delete node;

}

break;

}

if (market\_map[node->x][node->y-1] != '1' && visit[node->x][node->y-1] == 0) {

visit[node->x][node->y-1] = 1;

if (abs(abs(node->x - merchant\_address.x) - x\_difference) > 4 || abs(abs(node->y-1 - merchant\_address.y) - y\_difference) > 4) ;

else {

path\_node \*p = new path\_node;

\*p = {node->x, node->y-1};

p->front\_node = node;

bfsnode.push(p);

}

}

if (market\_map[node->x-1][node->y] != '1' && visit[node->x-1][node->y] == 0){

visit[node->x-1][node->y] = 1;

if (abs(abs(node->x-1 - merchant\_address.x) - x\_difference) > 4 || abs(abs(node->y - merchant\_address.y) - y\_difference) > 4) ;

else {

path\_node \*p = new path\_node;

\*p = {node->x-1, node->y};

p->front\_node = node;

bfsnode.push(p);

}

}

if (market\_map[node->x][node->y+1] != '1' && visit[node->x][node->y+1] == 0){

visit[node->x][node->y+1] = 1;

if (abs(abs(node->x - merchant\_address.x) - x\_difference) > 4 || abs(abs(node->y+1 - merchant\_address.y) - y\_difference) > 4) ;

else {

path\_node \*p = new path\_node;

\*p = {node->x, node->y+1};

p->front\_node = node;

bfsnode.push(p);

}

}

if (market\_map[node->x+1][node->y] != '1' && visit[node->x+1][node->y] == 0){

visit[node->x+1][node->y] = 1;

if (abs(abs(node->x+1 - merchant\_address.x) - x\_difference) > 4 || abs(abs(node->y - merchant\_address.y) - y\_difference) > 4) ;

else {

path\_node \*p = new path\_node;

\*p = {node->x+1, node->y};

p->front\_node = node;

bfsnode.push(p);

}

}

if (flag)

memory\_management.push(node);

flag = true;

}

// printf("搜索%d次\n", n);

while (!path.empty()) {

path\_node \*node = path.top();

// printf("(%d, %d)\n", node->x, node->y);

market\_map[node->x][node->y] = '+';

path.pop();

}

//释放内存2

while (!memory\_management.empty()) {

path\_node \*node = memory\_management.top();

memory\_management.pop();

delete node;

}

}

// Map.hpp

#ifndef Map\_hpp

#define Map\_hpp

#include <iostream>

#include <fstream>

#include <stack>

#include <queue>

#include <cmath>

#include "Global.hpp"

using namespace std;

//商家坐标

typedef struct {

int x;

int y;

string shop\_name;

}MapNode;

//路径

typedef struct path\_node{

int x;

int y;

struct path\_node \*front\_node;

}path\_node;

//地图类

class Map {

public:

Map(); //默认构造函数

void InitMap(); //读取文件建立地图

void SaveMap(); //写入文件保存地图

void ShowMap(); //输出地图

MapNode SearchMap(string &shop\_name); //根据商家店名搜索商家，返回该商家所在的坐标

void ChangeMap(const MapNode &merchant\_address, path\_node &start); //广度优先搜索与剪枝建立画出最短路径

path\_node locate(char x, char y); //根据地图坐标返回在数组中的位置

bool AddAddress(string &shop\_name); //商家入驻时改变坐标，若商家已满，返回false

private:

char market\_map[MAP\_WIDTH][MAP\_LENGTH]; //地图

MapNode merchant[ADDRESS\_MAX\_NUM]; //商家坐标信息

int num; //当前商家数量

};

#endif /\* Map\_hpp \*/

// Global.cpp

#include "Global.hpp"

//清空缓存区

void MY\_FLUSH() {

char ch;

while((ch = getchar()) != '\n' && ch != EOF);

}

//按任意键返回

void MY\_PAUSE() {

MY\_FLUSH();

cout <<endl<< " 按任意键返回..." << endl;

getchar();

}

//

// Global.hpp

#ifndef Global\_hpp

#define Global\_hpp

#include <iostream>

#include <string>

using namespace std;

//关于哈希表

const int MAXSIZE = 500; //哈希表长度

const int HASH\_MAX = 20; //哈希函数阈值

//关于管理员

const string ADMIN\_PASSERWORD = "12345"; //管理员密码

//关于花卉市场地图

const int MAP\_LENGTH = 29; //花卉市场长度

const int MAP\_WIDTH = 22; //花卉市场宽度

const int ADDRESS\_MAX\_NUM = 35; //花卉市场最大商家容量

//关于文件

const string FILE\_ADDRESS = "/Users/xumingfei/Desktop/git/Flower-market-management-system/基于青岛市枯桃花卉市场的交易管理系统/TXT/"; //文件所在地址

const string FILE\_TYPE = ".txt"; //文件类型

const string FLOWER\_FILE\_NAME = FILE\_ADDRESS + "data.txt"; //存放花卉信息的文件名

const string MERCHANT\_FILE\_NAME = FILE\_ADDRESS + "merchant.txt"; //存放商家账号密码和店名的文件名

const string PURCHASER\_FILE\_NAME = FILE\_ADDRESS + "purchaser.txt"; //存放买家账号密码的文件名

const string LEFT\_AD\_FILE\_NAME = FILE\_ADDRESS + "leftad.txt"; //存放左栏广告信息的文件名

const string RIGHT\_AD\_FILE\_NAME = FILE\_ADDRESS + "rightad.txt"; //存放右栏广告信息的文件名

const string ANNOUNCEMENT\_FILE\_NAME = FILE\_ADDRESS + "announcement.txt"; //存放公告信息的文件名

const string MAP\_FILE\_NAME = FILE\_ADDRESS + "map.txt"; //存放地图信息的文件名

const string MERCHANT\_ADDRESS\_FILE\_NAME = FILE\_ADDRESS + "address.txt"; //存放地图买家信息的文件名

//关于菜单

const string STAR = "★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★";

const string LHEAD = "╔═════════════";

const string HEAD = "═════════════════════";

const string RHEAD = "═════════════╗";

const string LBOTM = "╚═════════════";

const string BOTM = "═════════════════════";

const string RBOTM = "═════════════╝";

const string TT = " ";

const string T = " ";

const string FF = "║";

const string KK = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

const string MM = " ";

const string KONGGE = " ";

const string KKK = "　　　　　　　　 ";

const string SOFTWARE = "青岛市枯桃花卉市场交易管理系统";

const string AUTHOR = "Dousir9";

const string VERSION = "1.0";

const string NEW\_LINE = "\n";

//清空缓存区

void MY\_FLUSH();

//按任意键返回

void MY\_PAUSE();

#endif /\* Global\_hpp \*/

// Hash.cpp

#include "Hash.hpp"

//默认构造函数

HashTable::HashTable() : first(MAXSIZE) {

length = 0; //哈希表长度初始化为0

for (int i = 0; i < 500; i++) {

first[i] = NULL; //将哈希表每个位置的结点指针都指向NULL

}

}

//哈希函数,返回一个哈希表的下标

const int HashTable::Hash(const string &flower\_name) const {

int frt, sed, trd, sum;

frt = flower\_name[0];

sed = flower\_name[1];

trd = flower\_name[2];

frt = abs(frt)%10;

sed = abs(sed)%10;

trd = abs(trd)%10;

sum = ((frt+sed+trd)%10\*83+frt\*100+sed\*10+trd)%500;

return sum;

}

//同名不同商家的花卉采用链地址法处理冲突

void HashTable::AddChain(int index, LNode\* &p) {

LNode \*r = first[index];

p->next = r->next;

r->next = p;

}

//商家下架花卉时对同名花卉删除

bool HashTable::DeleteNode(int index, string &shop\_name) {

LNode \*r = first[index];

if(r->flower->ShopNameInfo() == shop\_name) {

first[index] = r->next;

delete r;

return true;

}

while(r->next) {

if(r->next->flower->ShopNameInfo() == shop\_name) {

LNode \*p = r->next;

r->next = p->next;

delete p;

return true;

}

r = r->next;

}

return false;

}

//不同命花卉二次探测法处理处理冲突

const int HashTable::DetectSecond(const int index, int times) const {

int detection;

int k = 1;

for (int i = 1; i <= times; i++) {

if (i % 2 == 1)

detection = k \* k;

else

detection = -k \* k;

if (i % 2 == 0)

++k;

}

if (detection > 0)

return (index + detection) % MAXSIZE;

else if (index + detection >= 0)

return index + detection;

else {

int sum = 500;

while (sum < -detection - index) {

sum += 500;

}

return sum - (-detection - index);

// 500 20 21 499

// 1000 20 521 499

}

}

//读取文件建立哈希表

void HashTable::InitHashTable() {

ifstream OutMyFile;

OutMyFile.open(FLOWER\_FILE\_NAME);

while (!OutMyFile.eof()) {

LNode \*p = new LNode;

string flower\_name; //花卉名称

string shop\_name; //花卉所在的店名

string petal\_color; //花卉花瓣颜色

float flower\_price; //花卉价格

int flower\_number; //花卉数量

int year, month, day; //花卉上架日期

OutMyFile>>flower\_name;

OutMyFile>>shop\_name;

OutMyFile>>petal\_color;

OutMyFile>>flower\_price;

OutMyFile>>flower\_number;

OutMyFile>>year;

OutMyFile>>month;

OutMyFile>>day;

p->flower = new Flower(flower\_name, shop\_name, petal\_color, flower\_price, flower\_number, year, month, day);

AddFlower(p, 1);

}

OutMyFile.close();

}

//建立哈希表时向哈希表添加花卉

void HashTable::AddFlower(LNode\* &p, int times) {

int index = Hash(p->flower->FlowerNameInfo());

if (first[index] == NULL) {

first[index] = p;

}

//同名不同商家花卉链地址处理法处理冲突

else if (first[index]->flower->FlowerNameInfo() == p->flower->FlowerNameInfo()) {

AddChain(index, p);

}

//不同名花卉二次探测法处理冲突

else {

index = DetectSecond(index, times);

if (first[index] == NULL) {

first[index] = p;

}

//同名不同商家花卉链地址处理法处理冲突

else if (first[index]->flower->FlowerNameInfo() == p->flower->FlowerNameInfo()) {

AddChain(index, p);

}

//探测次数加1，再次探测

else {

++times;

if (times == HASH\_MAX) { //哈希函数阈值

cout << "需重写哈希函数" << endl;

exit(0);

}

AddFlower(p, times);

}

}

}

//下架花卉

bool HashTable::DeleteFlower(string &flower\_name, string &shop\_name, int times) {

int index = Hash(flower\_name);

if (first[index] == NULL) {

return false;

}

if (first[index]->flower->FlowerNameInfo() == flower\_name) {

return DeleteNode(index, shop\_name);

}

//不同名花卉二次探测法处理冲突

else {

index = DetectSecond(index, times);

//同名不同商家花卉链地址处理法处理冲突

if (first[index]->flower->FlowerNameInfo() == flower\_name) {

return DeleteNode(index, shop\_name);

}

//探测次数加1，再次探测

else {

++times;

return DeleteFlower(flower\_name, shop\_name, times);

}

}

}

//在哈希表查找花卉并返回下标

int HashTable::SearchFlower(string &flower\_name, int times) {

int index = Hash(flower\_name);

if (first[index] == NULL)

return -1;

if (first[index]->flower->FlowerNameInfo() == flower\_name) {

return index;

}

//不同名花卉二次探测法处理冲突

else {

index = DetectSecond(index, times);

//同名不同商家花卉链地址处理法处理冲突

if (first[index]->flower->FlowerNameInfo() == flower\_name) {

return index;

}

//探测次数加1，再次探测

else {

++times;

if (times == HASH\_MAX) //若不存在或者哈希函数不够好返回-1

return -1;

return SearchFlower(flower\_name, times);

}

}

}

//该函数在哈希表的析构函数中，用于程序主动退出或意外结束时进行保存

void HashTable::SaveHashTable() {

bool flag = false; //控制写入文件的开头时是否换行

ofstream InMyFile;

InMyFile.open(FLOWER\_FILE\_NAME);

for (int i = 0; i < MAXSIZE; ++i) {

LNode \*p = first[i];

while (p != NULL) {

if (flag == true)

InMyFile<<endl;

flag = true;

InMyFile<<p->flower->FlowerNameInfo()<<" "<<p->flower->ShopNameInfo()<<" "<<p->flower->PetalColorInfo()<<" "<<p->flower->FlowerPriceInfo()<<" "<<p->flower->FlowerNumberInfo()<<" "<<p->flower->YearInfo()<<" "<<p->flower->MonthInfo()<<" "<<p->flower->DayInfo();

p = p->next;

}

}

InMyFile.close();

}

HashTable::~HashTable() {

SaveHashTable();

}

// Hash.hpp

#ifndef Hash\_hpp

#define Hash\_hpp

#include <iostream>

#include <vector>

#include <cmath>

#include <fstream>

#include "Flower.hpp"

#include "Global.hpp"

using namespace std;

//哈希表链地址处理法结点

typedef struct node{

node() { next = NULL; };

Flower \*flower;

struct node \*next;

}LNode;

//哈希表类

class HashTable {

public:

HashTable(); //默认构造函数;

void InitHashTable(); //读取文件信息建立哈希表

const int Hash(const string &name) const; //哈希函数，返回一个哈希表的下标

void AddFlower(LNode\* &p, int times); //建立哈希表时向哈希表添加花卉,time为二次探测法探测次数

void AddChain(int index, LNode\* &p); //同名不同商家的花卉采用链地址法处理冲突

bool DeleteFlower(string &flower\_name, string &shop\_name, int times); //下架花卉

bool DeleteNode(int index, string &shop\_name); //商家下架花卉时对同名花卉删除

int SearchFlower(string &flower\_name, int times); //在哈希表查找花卉并返回下标

const int DetectSecond(const int index, int times) const; //不同名的花卉采用二次探测法处理冲突,time为二次探测法探测次数

void SaveHashTable(); //程序结束时析构函数调用该函数将信息写回文件，实现信息保存

~HashTable(); //析构函数

vector<LNode\*> first; //指向花结点

private:

int length;

};

#endif /\* Hash\_hpp \*/

// Flower.cpp/

#include "Flower.hpp"

//默认构造函数

Flower::Flower() {

flower\_name = "";

shop\_name = "";

flower\_color = "";

flower\_price = 0;

flower\_number = 0;

//flower\_date调用默认构造函数

}

//进行初始化的构造函数

Flower::Flower(string &f\_name, string &s\_name, string &f\_color, float f\_price, int f\_number, int yy, int mm, int dd) : flower\_date(yy, mm ,dd) {

flower\_name = f\_name;

shop\_name = s\_name;

flower\_color = f\_color;

flower\_price = f\_price;

flower\_number = f\_number;

}

//设置花卉信息

void Flower::SetFlowerInfo(string &f\_name, string &s\_name, string &f\_color, float f\_price, int f\_number, int yy, int mm, int dd) {

flower\_name = f\_name;

shop\_name = s\_name;

flower\_color = f\_color;

flower\_price = f\_price;

flower\_number = f\_number;

flower\_date.SetDateInfo(yy, mm, dd);

}

//修改花卉价格

void Flower::ModifyFlowerPrice(const float &f\_price) {

flower\_price = f\_price;

}

//修改花卉数量

void Flower::ModifyFlowerNumber(const int &f\_number) {

flower\_number = f\_number;

}

//修改花卉上架日期

void Flower::ModifyFlowerDate(int yy, int mm, int dd) {

flower\_date.SetDateInfo(yy, mm, dd);

}

//修改花卉的花瓣颜色

void Flower::ModifyPetalColor(const string &f\_color) {

flower\_color = f\_color;

}

//返回花卉名称

const string& Flower::FlowerNameInfo() const {

return flower\_name;

}

//返回花卉所在的店名

const string& Flower::ShopNameInfo() const {

return shop\_name;

}

//返回花卉价格

const float& Flower::FlowerPriceInfo() const {

return flower\_price;

}

//返回花卉的花瓣颜色

const string& Flower::PetalColorInfo() const {

return flower\_color;

}

//返回花卉数量

const int& Flower::FlowerNumberInfo() const {

return flower\_number;

}

//返回年份信息

const int& Flower::YearInfo() {

return flower\_date.YearInfo();

}

//返回月份信息

const int& Flower::MonthInfo() {

return flower\_date.MonthInfo();

}

//返回年份信息

const int& Flower::DayInfo() {

return flower\_date.DayInfo();

}

//返回花卉上架日期

const FlowerDate& Flower::FlowerDateInfo() const {

return flower\_date;

}

// Flower.hpp

#ifndef Flower\_hpp

#define Flower\_hpp

#include <iostream>

#include <string>

#include "FlowerDate.hpp"

using namespace std;

//花类

class Flower {

public:

Flower(); //默认构造函数

Flower(string &f\_name, string &s\_name, string &f\_color, float f\_price, int f\_number, int yy, int mm, int dd);//进行初始化的构造函数

void SetFlowerInfo(string &f\_name, string &s\_name, string &f\_color, float f\_price, int f\_number, int yy, int mm, int dd); //设置花卉信息

void ModifyFlowerPrice(const float &f\_price); //修改花卉价格

void ModifyFlowerNumber(const int &f\_number); //修改花卉数量

void ModifyFlowerDate(int yy, int mm, int dd); //修改花卉上架日期

void ModifyPetalColor(const string &f\_color); //修改花卉的花瓣颜色

const string& FlowerNameInfo() const; //返回花卉名称

const string& ShopNameInfo() const; //返回花卉所在的店名

const string& PetalColorInfo() const; //返回花卉的花瓣颜色

const float& FlowerPriceInfo() const; //返回花卉价格

const int& FlowerNumberInfo() const; //返回花卉数量

const FlowerDate& FlowerDateInfo() const; //返回花卉上架日期

const int& YearInfo(); //返回年份信息

const int& MonthInfo(); //返回月份信息

const int& DayInfo(); //返回日信息

private:

string flower\_name; //花卉名称

string shop\_name; //花卉所在的店名

string flower\_color; //花卉花瓣颜色

float flower\_price; //花卉价格

int flower\_number; //花卉数量

FlowerDate flower\_date; //花卉上架日期

};

#endif /\* Flower\_hpp \*/

// FlowerDate.cpp

#include "FlowerDate.hpp"

//默认构造函数

FlowerDate::FlowerDate() {

year = 0;

month = 0;

day = 0;

}

//进行初始化的构造函数

FlowerDate::FlowerDate(int year, int month, int day) {

this->year = year;

this->month = month;

this->day = day;

}

//复制构造函数

FlowerDate::FlowerDate(const FlowerDate &flower\_date) {

year = flower\_date.year;

month = flower\_date.month;

day = flower\_date.day;

}

//设置日期函数

void FlowerDate::SetDateInfo(int year, int month, int day) {

this->year = year;

this->month = month;

this->day = day;

}

//返回年份信息

const int& FlowerDate::YearInfo() {

return year;

}

//返回月份信息

const int& FlowerDate::MonthInfo() {

return month;

}

//返回年份信息

const int& FlowerDate::DayInfo() {

return day;

}

//重载<<运算符

ostream& operator << (ostream &out, FlowerDate flower\_date) {

out<<left<<setw(2)<<flower\_date.YearInfo()<<"/"<<left<<setw(2)<<flower\_date.MonthInfo()<<"/"<<left<<setw(2)<<flower\_date.DayInfo()<<endl;

return out;

}

// FlowerDate.hpp

#ifndef FlowerDate\_hpp

#define FlowerDate\_hpp

#include <iostream>

#include <fstream>

#include <iomanip>

using namespace std;

//花卉上架日期类

class FlowerDate {

public:

FlowerDate(); //默认构造函数

FlowerDate(int year, int month, int day); //进行初始化的构造函数

FlowerDate(const FlowerDate &flower\_date); //复制构造函数

void SetDateInfo(int year, int month, int day); //设置日期函数

const int& YearInfo(); //返回年份信息

const int& MonthInfo(); //返回月份信息

const int& DayInfo(); //返回日信息

friend ostream& operator << (ostream &out, FlowerDate flower\_date);

private:

int year; //花卉上架的年份

int month; //花卉上架的月份

int day; //花卉上架的日

};

#endif /\* FlowerDate\_hpp \*/

// User.cpp

#include "User.hpp"

//默认构造函数

User::User() {

user\_name = "";

password = "";

}

//进行初始化的构造函数

User::User(const string &user\_name, const string &password) {

this->user\_name = user\_name;

this->password = password;

}

//返回用户名，用于对应买家文件名

const string& User::UserNameInfo() const {

return user\_name;

}

//修改登录状态

void User::ModifyLogStatus(bool judge) {

log\_success = judge;

}

//是否登录成功，登录成功返回true，失败返回false

bool User::IsLogSuccess() {

return log\_success;

}

// User.hpp

#ifndef User\_hpp

#define User\_hpp

#include <iostream>

#include <string>

#include <fstream>

#include "Global.hpp"

#include "Flower.hpp"

#include "Hash.hpp"

using namespace std;

//用户类

class User {

public:

User(); //默认构造函数

User(const string &user\_name, const string &password); //构造函数

const string& UserNameInfo() const; //返回返回用户名，用于对应买家文件名

void ModifyLogStatus(bool judge); //修改登录状态

bool IsLogSuccess(); //是否登录成功，登录成功返回true，失败返回false

protected:

bool log\_success; //判断是否登录成功

private:

string user\_name; //用户名

string password; //密码

};

#endif /\* User\_hpp \*/

// Menu.cpp

#include "Menu.hpp"

#include "Hash.hpp"

extern HashTable hashtable;

extern char leftad1[30], leftad2[30], leftad3[30], leftad4[30], leftad5[30], leftad6[30], leftad7[30], leftad8[30], leftad9[30], leftad10[30];

extern char rightad1[30], rightad2[30], rightad3[30], rightad4[30], rightad5[30], rightad6[30], rightad7[30], rightad8[30], rightad9[30], rightad10[30];

extern char announcement1[100], announcement2[100], announcement3[100], announcement4[100], announcement5[100];

//基类====================================================================

//默认构造函数

MainMenu::MainMenu() { }

//输出菜单头部

void MainMenu::ShowMenuHead() {

cout<<endl; cout<<" ";

cout<<STAR<<" "<<SOFTWARE<<" "<<STAR<<NEW\_LINE<<NEW\_LINE;

}

//输出菜单底部

void MainMenu::ShowMenuBottom() {

cout << "\n\n 【公告】 \n";

usleep(40000);

cout << " ┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┓\n";

usleep(40000);

cout << " {@}"; ShowAnnouncement(announcement1); cout << " {@}\n";

usleep(40000);

cout << " | "; ShowAnnouncement(announcement2); cout << " |\n";

usleep(40000);

cout << " \\|/"; ShowAnnouncement(announcement3); cout << " \\|/\n";

usleep(40000);

cout << " | "; ShowAnnouncement(announcement4); cout << " |\n";

usleep(40000);

cout << " | "; ShowAnnouncement(announcement5); cout << " |\n";

usleep(40000);

cout << " ┗━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┛\n";

usleep(40000);

cout << " :";

usleep(40000);

}

//输出版本信息

void MainMenu::ShowVersion() {

cout<<endl; cout<<" ";

cout<<STAR<<" "<<SOFTWARE<<" "<<STAR<<NEW\_LINE<<NEW\_LINE;

cout<<endl<<" 版本信息： V1.0"<<endl<<endl;

cout<<endl<<" 如果您发现该系统在某些地方,存在bug或者可以优化"<<endl<<

" 请联系我，我的QQ是: 736191200"<<endl<<endl;

cout<<endl<<" 如果您想继续完善该系统或者想进行有关内容的学习"<<endl<<

" 可以访问该系统的github网站进行fork或者查看"<<endl<<endl;

cout<<" github: https://github.com/Dousir9/Flower-market-management-system"<<endl;

cout<<endl<<endl<<endl<<endl<<endl<<endl<<" Copyright c 2019 Dousir9. All rights reserved."<<endl<<endl;

cout<<endl<<"\n ";

MY\_PAUSE();

}

//输出错误信息

void MainMenu::ShowError() {

cout << "\n\n\n\n\n\n\n\n\n\n\n\n\n ";

cout << " 无效的输入，请重新输入" << endl;

usleep(1500000);

system("clear");

}

//接收指令

char MainMenu::Input() {

char ch;

cin>>ch;

MY\_FLUSH();

return ch;

}

//加载广告

void MainMenu::LoadAd() {

ifstream LeftAd, RightAd;

LeftAd.open(LEFT\_AD\_FILE\_NAME);

RightAd.open(RIGHT\_AD\_FILE\_NAME);

int i = 0;

while (!LeftAd.eof()) {

if (i < 24)

leftad1[i%24] = LeftAd.get();

else if (i >= 24 && i < 48)

leftad2[i%24] = LeftAd.get();

else if (i >= 48 && i < 72)

leftad3[i%24] = LeftAd.get();

else if (i >= 72 && i < 96)

leftad4[i%24] = LeftAd.get();

else if (i >= 96 && i < 120)

leftad5[i%24] = LeftAd.get();

else if (i >= 120 && i < 144)

leftad6[i%24] = LeftAd.get();

else if (i >= 144 && i < 168)

leftad7[i%24] = LeftAd.get();

else if (i >= 168 && i < 192)

leftad8[i%24] = LeftAd.get();

else if (i >= 192 && i < 216)

leftad9[i%24] = LeftAd.get();

else if (i >= 216 && i < 240)

leftad10[i%24] = LeftAd.get();

++i;

if (i == 240)

break;

}

i = 0;

while (!RightAd.eof()) {

if (i < 24)

rightad1[i%24] = RightAd.get();

else if (i >= 24 && i < 48)

rightad2[i%24] = RightAd.get();

else if (i >= 48 && i < 72)

rightad3[i%24] = RightAd.get();

else if (i >= 72 && i < 96)

rightad4[i%24] = RightAd.get();

else if (i >= 96 && i < 120)

rightad5[i%24] = RightAd.get();

else if (i >= 120 && i < 144)

rightad6[i%24] = RightAd.get();

else if (i >= 144 && i < 168)

rightad7[i%24] = RightAd.get();

else if (i >= 168 && i < 192)

rightad8[i%24] = RightAd.get();

else if (i >= 192 && i < 216)

rightad9[i%24] = RightAd.get();

else if (i >= 216 && i < 240)

rightad10[i%24] = RightAd.get();

++i;

if (i == 240)

break;

}

}

//逐行输出广告

void MainMenu::ShowAd(char ad[]) {

cout << "\* ";

for (int i = 0; i < 24; i += 3) {

if (ad[i] != '\0' && ad[i+1] != '\0' && ad[i+2] != '\0')

cout << ad[i] << ad[i+1] << ad[i+2];

else if (i % 3 == 0)

cout << "　";

}

cout << " \*";

}

//加载公告

void MainMenu::LoadAnnouncement() {

ifstream Announcement;

Announcement.open(ANNOUNCEMENT\_FILE\_NAME);

int i = 0;

while (!Announcement.eof()) {

if (i < 90)

announcement1[i%90] = Announcement.get();

else if (i >= 90 && i < 180)

announcement2[i%90] = Announcement.get();

else if (i >= 180 && i < 270)

announcement3[i%90] = Announcement.get();

else if (i >= 270 && i < 360)

announcement4[i%90] = Announcement.get();

else if (i >= 360 && i < 450)

announcement5[i%90] = Announcement.get();

++i;

if (i == 450)

break;

}

}

//逐行输出公告

void MainMenu::ShowAnnouncement(char an[]){

cout<<(" ");

cout<<("┃ ");

for(int i = 0; i < 90; i += 3){

if(an[i] != '\0' && an[i+1] != '\0' && an[i+2] != '\0')

cout << an[i] << an[i+1] << an[i+2];

else if (i % 3 == 0)

cout << "　";

}

cout<<(" ┃");

}

//输出主菜单信息

void MainMenu::ShowMenuInfo() {

usleep(40000);

cout<<TT;cout<<KK; cout<<(" ╔════════════════════════════════════════════════╗ ");cout<<KK; cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad1); cout<<" ║ ① 买家入口 ║ "; ShowAd(rightad1); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad2); cout<<" ║ ║ ";ShowAd(rightad2); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad3); cout<<" ║ ② 商家入口 ║ "; ShowAd(rightad3); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad4); cout<<" ║ ║ ";ShowAd(rightad4); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad5); cout<<" ║ ③ 管理员入口 ║ "; ShowAd(rightad5); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad6); cout<<" ║ ║ ";ShowAd(rightad6); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad7); cout<<" ║ ④ 版本信息　 ║ "; ShowAd(rightad7); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad8); cout<<" ║ ║ ";ShowAd(rightad8); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad9); cout<<" ║ ⑤ 退出系统 ║ "; ShowAd(rightad9); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad10); cout<<" ║ 请按提示进行输入: ║ ";ShowAd(rightad10); cout<<endl;

usleep(40000);

cout<<TT;cout<<KK; cout<<(" ╚════════════════════════════════════════════════╝ ");cout<<KK; cout<<endl;

usleep(40000);

}

//主菜单运行

void MainMenu::MenuRun() {

system("clear");

ShowMenuHead();

ShowMenuInfo();

ShowMenuBottom();

char ch = Input();

static PurchaserLogMenu purchaser\_log\_menu;

static MerchantLogMenu merchant\_log\_menu;

static AdminLogMenu admin\_log\_menu;

switch (ch) {

case '1': //买家登录、注册入口

system("clear");

purchaser\_log\_menu.MenuRun();

MenuRun();

break;

case '2': //商家入驻、登录入口

system("clear");

merchant\_log\_menu.MenuRun();

MenuRun();

break;

case '3': //管理员入口

system("clear");

admin\_log\_menu.MenuRun();

MenuRun();

break;

case '4': //版本信息

system("clear");

ShowVersion();

MenuRun();

case '5': //退出系统

exit(0);

default: //应对异常输入

system("clear");

ShowError();

MenuRun();

}

}

//基类====================================================================

//商家====================================================================

//输出商家入驻、登录菜单

void MerchantLogMenu::ShowMenuInfo() {

usleep(40000);

cout<<TT;cout<<KK; cout<<" ╔════════════════════════════════════════════════╗ ";cout<<KK; cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad1); cout<<" ║ ║ "; ShowAd(rightad1); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad2); cout<<" ║ ① 商家登录 ║ ";ShowAd(rightad2); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad3); cout<<" ║ ║ "; ShowAd(rightad3); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad4); cout<<" ║ ② 商家入驻 ║ ";ShowAd(rightad4); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad5); cout<<" ║ ║ "; ShowAd(rightad5); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad6); cout<<" ║ ③ 返回 ║ ";ShowAd(rightad6); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad7); cout<<" ║ ║ "; ShowAd(rightad7); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad8); cout<<" ║ ④ 退出系统 ║ ";ShowAd(rightad8); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad9); cout<<" ║ ║ "; ShowAd(rightad9); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad10); cout<<" ║ 请按提示进行输入: ║ ";ShowAd(rightad10); cout<<endl;

usleep(40000);

cout<<TT;cout<<KK; cout<<" ╚════════════════════════════════════════════════╝ "; cout<<KK; cout<<endl;

usleep(40000);

}

//商家入驻、登录菜单运行

void MerchantLogMenu::MenuRun() {

system("clear");

ShowMenuHead();

ShowMenuInfo();

ShowMenuBottom();

char ch = Input();

static Merchant merchant;

static MerchantDoMenu merchant\_do\_menu;

switch (ch) {

case '1': //商家登录

system("clear");

merchant = merchant.Loggin();

if (merchant.IsLogSuccess())

merchant\_do\_menu.MenuRun(merchant);

MenuRun();

break;

case '2': //商家入驻

system("clear");

merchant.Logging();

MenuRun();

break;

case '3': //返回主菜单

return ;

break;

case '4': //退出系统

exit(0);

default: //应对异常输入

system("clear");

ShowError();

MenuRun();

}

}

//输出商家操作菜单

void MerchantDoMenu::ShowMenuInfo() {

usleep(40000);

cout<<TT;cout<<KK; cout<<" ╔════════════════════════════════════════════════╗ "; cout<<KK; cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad1); cout<<" ║ ① 查看所有花卉信息 ║ ";ShowAd(rightad1); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad2); cout<<" ║ ② 花卉上架 ║ ";ShowAd(rightad2); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad3); cout<<" ║ ③ 花卉下架 ║ ";ShowAd(rightad3); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad4); cout<<" ║ ④ 花卉信息修改 ║ ";ShowAd(rightad4); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad5); cout<<" ║ ⑤ 花卉自动折扣 ║ "; ShowAd(rightad5); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad6); cout<<" ║ ⑥ 返回 ║ ";ShowAd(rightad6); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad7); cout<<" ║ ⑦ 退出系统 ║ "; ShowAd(rightad7); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad8); cout<<" ║ 　　 ║ ";ShowAd(rightad8); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad9); cout<<" ║ 请按提示进行输入: ║ "; ShowAd(rightad9); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad10);cout<<" ║ ║ ";ShowAd(rightad10); cout<<endl;

usleep(40000);

cout<<TT;cout<<KK; cout<<" ╚════════════════════════════════════════════════╝ ";cout<<KK; cout<<endl;

usleep(40000);

}

//商家操作菜单运行

void MerchantDoMenu::MenuRun(const Merchant &merchant) {

system("clear");

ShowMenuHead();

ShowMenuInfo();

ShowMenuBottom();

char ch = Input();

switch (ch) {

case '1': //查看上架花卉

system("clear");

merchant.CheckFlower();

MY\_PAUSE();

MenuRun(merchant);

break;

case '2': //花卉上架

system("clear");

merchant.AddFlower();

MY\_PAUSE();

MenuRun(merchant);

break;

case '3': //花卉下架

system("clear");

merchant.DeleteFlower();

MY\_PAUSE();

MenuRun(merchant);

break;

case '4': //花卉信息修改

system("clear");

merchant.ModifyFlowerInfo();

MY\_PAUSE();

MenuRun(merchant);

break;

case '5': //花卉自动折扣

system("clear");

merchant.AutoDiscountFlower();

MY\_PAUSE();

MenuRun(merchant);

break;

case '6': //返回上级菜单

return ;

break;

case '7': //退出系统

exit(0);

default: //应对异常输入

system("clear");

ShowError();

MenuRun(merchant);

}

}

//商家====================================================================

//买家====================================================================

//输出买家注册、登录菜单

void PurchaserLogMenu::ShowMenuInfo() {

usleep(40000);

cout<<TT;cout<<KK; cout<<" ╔════════════════════════════════════════════════╗ ";cout<<KK; cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad1); cout<<" ║ ║ "; ShowAd(rightad1); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad2); cout<<" ║ ① 登录 ║ ";ShowAd(rightad2); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad3); cout<<" ║ ║ "; ShowAd(rightad3); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad4); cout<<" ║ ② 注册 ║ ";ShowAd(rightad4); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad5); cout<<" ║ ║ "; ShowAd(rightad5); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad6); cout<<" ║ ③ 返回 ║ ";ShowAd(rightad6); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad7); cout<<" ║ ║ "; ShowAd(rightad7); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad8); cout<<" ║ ④ 退出系统 ║ ";ShowAd(rightad8); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad9); cout<<" ║ ║ "; ShowAd(rightad9); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad10);cout<<" ║ 请按提示进行输入: ║ ";ShowAd(rightad10); cout<<endl;

usleep(40000);

cout<<TT;cout<<KK; cout<<" ╚════════════════════════════════════════════════╝ ";cout<<KK; cout<<endl;

usleep(40000);

}

//买家入驻、登录菜单运行

void PurchaserLogMenu::MenuRun() {

system("clear");

ShowMenuHead();

ShowMenuInfo();

ShowMenuBottom();

char ch = Input();

static Purchaser purchaser;

static PurchaserDoMenu purchaser\_do\_menu;

switch (ch) {

case '1': //买家登录

system("clear");

purchaser = purchaser.Loggin();

if (purchaser.IsLogSuccess())

purchaser\_do\_menu.MenuRun(purchaser);

MenuRun();

break;

case '2': //买家注册

system("clear");

purchaser.Logging();

MenuRun();

break;

case '3': //返回主菜单

return ;

break;

case '4': //退出系统

exit(0);

default: //应对异常输入

system("clear");

ShowError();

MenuRun();

}

}

//输出买家操作菜单

void PurchaserDoMenu::ShowMenuInfo() {

usleep(40000);

cout<<TT;cout<<KK; cout<<" ╔════════════════════════════════════════════════╗ ";cout<<KK; cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad1); cout<<" ║ ① 查看所有上架花卉 ║ "; ShowAd(rightad1); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad2); cout<<" ║ ② 花卉查找 ║ ";ShowAd(rightad2); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad3); cout<<" ║ ③ 花卉预约 ║ "; ShowAd(rightad3); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad4); cout<<" ║ ④ 查看预约 ║ ";ShowAd(rightad4); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad5); cout<<" ║ ⑤ 花卉推荐 ║ "; ShowAd(rightad5); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad6); cout<<" ║ ⑥ 花店地址导航 ║ ";ShowAd(rightad6); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad7); cout<<" ║ ⑦ 退出登录 ║ "; ShowAd(rightad7); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad8); cout<<" ║ ⑧ 退出系统 ║ ";ShowAd(rightad8); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad9); cout<<" ║ ║ "; ShowAd(rightad9); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad10);cout<<" ║ 请按提示进行输入: ║ ";ShowAd(rightad10); cout<<endl;

usleep(40000);

cout<<TT;cout<<KK; cout<<" ╚════════════════════════════════════════════════╝ ";cout<<KK; cout<<("\n");

usleep(40000);

}

//买家操作菜单运行

void PurchaserDoMenu::MenuRun(const Purchaser &purchaser) {

system("clear");

ShowMenuHead();

ShowMenuInfo();

ShowMenuBottom();

char ch = Input();

switch (ch) {

case '1': //查看所有上架花卉

system("clear");

purchaser.CheckALLFlower();

MY\_PAUSE();

MenuRun(purchaser);

break;

case '2': //查找花卉

system("clear");

purchaser.SearchFlower();

MY\_PAUSE();

MenuRun(purchaser);

break;

case '3': //花卉预约

system("clear");

purchaser.ReserveFlower();

MY\_PAUSE();

MenuRun(purchaser);

break;

case '4': //查看预约

system("clear");

purchaser.CheckReservation();

MY\_PAUSE();

MenuRun(purchaser);

break;

case '5': //花卉推荐

system("clear");

purchaser.RecommendFlower();

MY\_PAUSE();

MenuRun(purchaser);

break;

case '6': //花店地址导航

system("clear");

purchaser.NavigateShopAddress();

MY\_PAUSE();

MenuRun(purchaser);

break;

case '7': //返回上级菜单

return ;

break;

case '8': //退出系统

exit(0);

default: //应对异常输入

system("clear");

ShowError();

MenuRun(purchaser);

}

}

//买家====================================================================

//管理员===================================================================

//输出管理员登录菜单

void AdminLogMenu::ShowMenuInfo() {

cout <<endl<<endl<<endl;

}

//商家入驻、登录菜单运行

void AdminLogMenu::MenuRun() {

system("clear");

ShowMenuHead();

ShowMenuInfo();

static Admin admin;

static AdminDoMenu admin\_do\_menu;

if (admin.Loggin() == true)

admin\_do\_menu.MenuRun();

}

//输出管理员操作菜单

void AdminDoMenu::ShowMenuInfo() {

cout<<TT;cout<<KK; cout<<" ╔════════════════════════════════════════════════╗ ";cout<<KK; cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad1); cout<<" ║ ║ "; ShowAd(rightad1); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad2); cout<<" ║ ① 公告发布 ║ ";ShowAd(rightad2); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad3); cout<<" ║ ② 广告设置 ║ "; ShowAd(rightad3); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad4); cout<<" ║ ③ 广告一键清空 ║ ";ShowAd(rightad4); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad5); cout<<" ║ ④ 买家找回密码 ║ "; ShowAd(rightad5); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad6); cout<<" ║ ⑤ 商家找回密码 ║ ";ShowAd(rightad6); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad7); cout<<" ║ ⑥ 返回主菜单 ║ "; ShowAd(rightad7); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad8); cout<<" ║ ⑦ 退出系统 ║ ";ShowAd(rightad8); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad9); cout<<" ║ ║ "; ShowAd(rightad9); cout<<endl;

usleep(40000);

cout<<TT;ShowAd(leftad10);cout<<" ║ 请按提示进行输入: ║ ";ShowAd(rightad10); cout<<endl;

usleep(40000);

cout<<TT;cout<<KK; cout<<(" ╚════════════════════════════════════════════════╝ ");cout<<KK; cout<<endl;

usleep(40000);

}

//管理员操作菜单运行

void AdminDoMenu::MenuRun() {

system("clear");

ShowMenuHead();

ShowMenuInfo();

ShowMenuBottom();

static Admin admin;

char ch = Input();

switch (ch) {

case '1': //发布公告

system("clear");

admin.Announce();

LoadAnnouncement();

MY\_PAUSE();

MenuRun();

break;

case '2': //设置广告

system("clear");

admin.SetAd();

LoadAd();

MY\_PAUSE();

MenuRun();

break;

case '3': //广告一键清空

system("clear");

admin.ClearAd();

LoadAd();

MY\_PAUSE();

MenuRun();

break;

case '4': //买家找回密码

system("clear");

admin.PurchaserRetrievePassword();

MY\_PAUSE();

MenuRun();

break;

case '5': //商家找回密码

system("clear");

admin.MerchantRetrievePassword();

MY\_PAUSE();

MenuRun();

break;

case '6': //返回主菜单

return ;

case '7': //退出系统

exit(0);

default: //应对异常输入

system("clear");

ShowError();

MenuRun();

}

}

//管理员===================================================================

//

// Menu.hpp

#ifndef Menu\_hpp

#define Menu\_hpp

#include <iostream>

#include <cstdlib>

#include <unistd.h>

#include "Merchant.hpp"

#include "Purchaser.hpp"

#include "Admin.hpp"

//（主）菜单类

class MainMenu {

public:

MainMenu();

void LoadAd(); //加载广告

void LoadAnnouncement(); //加载公告

void ShowMenuHead(); //输出菜单头部

void ShowMenuInfo(); //输出菜单内容

void ShowMenuBottom(); //输出菜单底部

void ShowAd(char ad[]); //输出广告内容

void ShowAnnouncement(char an[]); //输出公告

void ShowError(); //输出异常信息

void ShowVersion(); //输出版本信息

void MenuRun(); //运行菜单

char Input(); //从键盘接收指令

};

//商家登录注册菜单类，继承菜单类

class MerchantLogMenu : public MainMenu {

public:

void ShowMenuInfo(); //输出菜单内容

void MenuRun(); //运行菜单

};

//商家操作菜单类，继承菜单类

class MerchantDoMenu : public MainMenu {

public:

void ShowMenuInfo(); //输出菜单内容

void MenuRun(const Merchant &merchant); //运行菜单

};

//买家登录注册菜单类，继承菜单类

class PurchaserLogMenu : public MainMenu {

public:

int extracted();

void ShowMenuInfo(); //输出菜单内容

void MenuRun(); //运行菜单

};

//买家操作菜单类，继承菜单类

class PurchaserDoMenu : public MainMenu {

public:

void ShowMenuInfo(); //输出菜单内容

void MenuRun(const Purchaser &purchaser); //运行菜单

};

//管理员登录菜单类，继承菜单类

class AdminLogMenu : public MainMenu {

public:

void ShowMenuInfo(); //输出菜单内容

void MenuRun(); //运行菜单

};

//管理员操作菜单类，继承菜单类

class AdminDoMenu : public MainMenu {

public:

void ShowMenuInfo(); //输出菜单内容

void MenuRun(); //运行菜单

};

#endif /\* Menu\_hpp \*/

// Merchant.cpp

#include "Merchant.hpp"

#include "Hash.hpp"

extern HashTable hashtable;

//默认构造函数

Merchant::Merchant() {

shop\_name = "";

log\_success = false;

}

//进行初始化的构造函数

Merchant::Merchant(const string &user\_name, const string &password, const string &shop\_name) : User(user\_name, password){

this->shop\_name = shop\_name;

log\_success = false;

}

//商家入驻

void Merchant::Logging() {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

string user\_name;

string password1, password2;

string shop\_name;

cout<<endl<<" 请输入注册用户名: ";

cin >> user\_name;

//验证用户名是否存在

ifstream InMyFile1;

InMyFile1.open(MERCHANT\_FILE\_NAME);

string file\_user\_name, file\_password, file\_shop\_name;

while (!InMyFile1.eof()) {

InMyFile1>>file\_user\_name>>file\_password>>file\_shop\_name;

if (file\_user\_name == user\_name) {

InMyFile1.close();

cout<<endl<<" 该用户名已被注册，入驻失败"<<endl;

MY\_PAUSE();

return ;

}

}

InMyFile1.close();

//输入两次密码

cout<<endl<<" 请输入密码(不小于8位): ";

cin >> password1;

if (password1.length() < 8) {

cout<<endl<<" 密码长度应不小于8位，入驻失败"<<endl;

MY\_PAUSE();

return ;

}

cout<<endl<<" 请再次输入密码: ";

cin >> password2;

if (password1 != password2) {

cout<<endl<<" 两次输入的密码不一致，入驻失败"<<endl;

MY\_PAUSE();

return ;

}

cout<<endl<<" 请输入花店名:(4字） ";

cin >> shop\_name;

//验证花店名已存在

ifstream InMyFile2;

InMyFile2.open(MERCHANT\_FILE\_NAME);

while (!InMyFile2.eof()) {

InMyFile2>>file\_user\_name>>file\_password>>file\_shop\_name;

if (file\_shop\_name == shop\_name) {

cout<<endl<<" 该店名已存在， 入驻失败"<<endl;

MY\_PAUSE();

return ;

}

}

InMyFile2.close();

Map m;

m.InitMap();

if (m.AddAddress(shop\_name) == true) {

m.SaveMap();

fstream InMyFile3;

InMyFile3.open(MERCHANT\_FILE\_NAME, ios::app);

InMyFile3.seekg(0, ios::end); //将文件指针指向文件末端

streampos fp = InMyFile3.tellg(); //fp为文件指针的偏移量

if (int(fp) != 0) // 偏移量为0，证明文件为空，为首次进入系统,不是首次进入系统就换行

InMyFile3<<endl;

InMyFile3<<user\_name<<" "<<password1<<" "<<shop\_name;

InMyFile3.close();

cout<<endl<<" 入驻成功"<<endl;

MY\_PAUSE();

} else {

cout<<endl<<" 市场已满入驻失败"<<endl;

MY\_PAUSE();

}

}

//商家登录

Merchant Merchant::Loggin() {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

string user\_name;

string password;

cout<<endl<<" 请 输 入 用 户 名:";

cin >> user\_name;

cout<<endl<<" 请 输 入 密 码:";

cin >> password;

string file\_user\_name; //文件中的用户名

string file\_password; //文件中的密码

string file\_shop\_name; //文件中的店名

ifstream OutMyFile;

OutMyFile.open(MERCHANT\_FILE\_NAME);

while (!OutMyFile.eof()) {

OutMyFile>>file\_user\_name>>file\_password>>file\_shop\_name;

if (file\_user\_name == user\_name) {

if (file\_password == password) {

OutMyFile.close();

cout<<endl<<" 密码正确,正在登录...."<<endl;

sleep(2);

Merchant m(user\_name, password, file\_shop\_name);

m.ModifyLogStatus(true);

return m;

} else {

OutMyFile.close();

cout<<endl<<" 密码错误，正在返回商家界面...."<<endl;

sleep(2);

Merchant m;

return m;

}

}

}

OutMyFile.close();

cout<<endl<<" 该用户名不存在，正在返回商家界面...."<<endl;

sleep(2);

Merchant m;

return m;

}

//查看自家上架花卉

void Merchant::CheckFlower() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

printf(" 花卉名称 花店名 花卉颜色 单价 数量 上架日期 \n");

for (int i = 0; i < MAXSIZE; ++i) {

LNode \*p = hashtable.first[i];

while (p != NULL) {

if(p->flower->ShopNameInfo() == shop\_name) {

printf(" %-8s %-8s %-5s %5.1f %-3d ", p->flower->FlowerNameInfo().c\_str(), p->flower->ShopNameInfo().c\_str(), p->flower->PetalColorInfo().c\_str(),p->flower->FlowerPriceInfo(), p->flower->FlowerNumberInfo());

//用到运算符重载 <<

cout<<p->flower->FlowerDateInfo();

}

p = p->next;

usleep(25000);

}

}

}

//花卉上架

void Merchant::AddFlower() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

string flower\_name;

string shop\_name = this->shop\_name;

string flower\_color;

float flower\_price;

int flower\_number;

int year, month, day;

cout << " 请输入花卉名称: ";

cin >> flower\_name;

cout << " 请输入花卉颜色: ";

cin >> flower\_color;

cout << " 请输入花卉价格: ";

cin >> flower\_price;

cout << " 请输入花卉数量: ";

cin >> flower\_number;

cout << " 请输入花卉上架日期(年 月 日): ";

cin >> year >> month >> day;

LNode \*p = new LNode;

p->flower = new Flower(flower\_name, shop\_name, flower\_color, flower\_price, flower\_number, year, month, day);

hashtable.AddFlower(p, 1);

cout <<endl<< " 花卉上架成功" << endl;

}

//花卉下架

void Merchant::DeleteFlower() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

string flower\_name;

string shop\_name = this->shop\_name;

cout << " 请输入你要下架花卉的名称 :";

cin >> flower\_name;

if (hashtable.DeleteFlower(flower\_name, shop\_name, 1))

cout <<endl<< " 花卉下架成功" <<endl;

else

cout <<endl<< " 要下架的花卉不存在" << endl;

}

//花卉信息修改

void Merchant::ModifyFlowerInfo() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

string flower\_name;

cout<<endl<<" 请输入你要修改的花卉名称: ";

cin >> flower\_name;

int index = hashtable.SearchFlower(flower\_name, 1);

if (index == -1) {

cout<<endl<<" 您要修改的花卉不存在"<<endl;

return ;

} else {

LNode \*p = hashtable.first[index];

while(p != NULL) {

if (p->flower->ShopNameInfo() == shop\_name) {

cout<<endl<<" 您要修改哪个信息: "<<endl;

cout<<endl<<" 1.花卉颜色 "<<endl;

cout<<endl<<" 2.花卉价格 "<<endl;

cout<<endl<<" 3.花卉数量 "<<endl;

cout<<endl<<" 4.花卉上架日期 "<<endl;

char ch;

string color;

float price;

int number;

int yy, mm, dd;

cin >> ch;

switch (ch) {

case '1':

cout << " 当前花卉颜色为: " << p->flower->PetalColorInfo() <<endl;

cout << " 请输入修改后的花卉颜色: ";

cin >> color;

fflush(stdin);

p->flower->ModifyPetalColor(color);

cout <<endl<<" "<< p->flower->FlowerNameInfo() <<"颜色修改成功" << endl;

break;

case '2':

cout << " 当前花卉价格为: " << p->flower->FlowerPriceInfo() <<endl;

cout << " 请输入要修改后的价格: ";

cin >> price;

fflush(stdin);

p->flower->ModifyFlowerPrice(price);

cout <<endl<<" "<< p->flower->FlowerNameInfo() <<"价格修改成功" << endl;

break;

case '3':

cout << " 当前花卉数量为: " << p->flower->FlowerNumberInfo() <<endl;

cout << " 请输入要修改后的数量: ";

cin >> number;

fflush(stdin);

p->flower->ModifyFlowerNumber(number);

cout <<endl<<" "<< p->flower->FlowerNameInfo() <<"数量修改成功" << endl;

break;

case '4':

cout << " 当前花卉上架日期为: " << p->flower->YearInfo() <<"/"<<p->flower->MonthInfo()<<"/"<<p->flower->DayInfo()<<endl;

cout << " 请输入要修改后的上架日期(yy mm dd): ";

cin >> yy >> mm >> dd;

fflush(stdin);

p->flower->ModifyFlowerDate(yy, mm, dd);

cout <<endl<<" "<< p->flower->FlowerNameInfo() <<"上架日期修改成功" << endl;

break;

default:

cout << " 无效的输入" << endl;

}

return ;

}

p = p->next;

}

}

cout << " 你要修改的花卉不存在" << endl;

}

//花卉自动折扣

void Merchant::AutoDiscountFlower() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

cout << " 请输入折扣花卉的已上架天数: ";

int num;

cin >> num;

int yy2, mm2, dd2;

time\_t timep;

struct tm \*p;

time(&timep);

p=gmtime(&timep);

yy2 = 1900+p->tm\_year;

mm2 = 1+p->tm\_mon;

dd2 = p->tm\_mday;

cout << " 请输入折扣(0.0——1.0): ";

float discount;

cin >> discount;

for (int i = 0; i < MAXSIZE; ++i) {

LNode \*p = hashtable.first[i];

while (p != NULL) {

if(p->flower->ShopNameInfo()==shop\_name && FlowerDateCompare(p->flower->YearInfo(), p->flower->MonthInfo(), p->flower->DayInfo(), yy2, mm2, dd2) >= num) {

cout <<endl<<" "<< p->flower->FlowerNameInfo() << "的价格从"<< p->flower->FlowerPriceInfo() << "变为";

p->flower->ModifyFlowerPrice(p->flower->FlowerPriceInfo() \* discount);

cout << p->flower->FlowerPriceInfo() <<endl;

usleep(25000);

}

p = p->next;

}

}

}

//比较日期，返回差值 (yy2/mm2/dd2 - yy1/mm1/dd1)

const int Merchant::FlowerDateCompare(int yy1, int mm1, int dd1, int yy2, int mm2, int dd2) const {

int sum = 0;

if (mm1==mm2) {

return dd2 - dd1;

}

else if (mm1 < mm2-1) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if ((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

while (mm1 < mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31;

break;

case 4:case 6:case 9:case 11:

sum += 30;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29;

else

sum += 28;

}

mm1++;

}

sum += dd2 - 1;

return sum;

}

else if (mm1 < mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

sum += dd2 - 1;

return sum;

}

else if (mm1 > mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

while (mm1 < 13) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31;

break;

case 4:case 6:case 9:case 11:

sum += 30;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29;

else

sum += 28;

}

mm1++;

}

mm1 = 1;

if(mm1 < mm2-1){

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if((yy2%4==0 && yy2%100!=0)||yy2%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

while(mm1 < mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31;

break;

case 4:case 6:case 9:case 11:

sum += 30;

break;

case 2:

if((yy2%4==0 && yy2%100!=0) || yy2%400==0)

sum += 29;

else

sum += 28;

}

sum += dd2 - 1;

return sum;

}

}

else if (mm1 < mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

sum += dd2 - 1;

return sum;

}

}

return 0;

}

// Merchant.hpp

#ifndef Merchant\_hpp

#define Merchant\_hpp

#include <iostream>

#include <string>

#include <fstream>

#include <ctime>

#include <cstdio>

#include <unistd.h>

#include "Global.hpp"

#include "Flower.hpp"

#include "Hash.hpp"

#include "User.hpp"

#include "Map.hpp"

using namespace std;

//商家类，继承用户类

class Merchant : virtual public User {

public:

Merchant(); //默认构造函数

Merchant(const string &user\_name, const string &password, const string &shop\_name);

void Logging(); //商家入驻

Merchant Loggin(); //商家登录,登陆成功返回true，失败返回false

void CheckFlower() const; //查看自家上架花卉

void AddFlower() const; //上架花卉

void DeleteFlower() const; //下架花卉

void ModifyFlowerInfo() const; //修改花卉信息

void AutoDiscountFlower() const; //花卉自动折扣

const int FlowerDateCompare(int yy1, int mm1, int dd1, int yy2, int mm2, int dd2) const;

string& ShopNameInfo(); //返回花店名信息

private:

string shop\_name; //花店名

};

#endif /\* Merchant\_hpp \*/

// Purchaser.cpp

#include "Purchaser.hpp"

#include "Hash.hpp"

extern HashTable hashtable;

extern char leftad1[30], leftad2[30], leftad3[30], leftad4[30], leftad5[30], leftad6[30], leftad7[30], leftad8[30], leftad9[30], leftad10[30];

extern char rightad1[30], rightad2[30], rightad3[30], rightad4[30], rightad5[30], rightad6[30], rightad7[30], rightad8[30], rightad9[30], rightad10[30];

extern char announcement1[100], announcement2[100], announcement3[100], announcement4[100], announcement5[100];

//默认构造函数

Purchaser::Purchaser() {

LinkList = NULL;

log\_success = false;

}

//进行初始化的构造函数

Purchaser::Purchaser(const string &user\_name, const string &password) : User(user\_name, password){

LinkList = NULL;

log\_success = false;

}

//买家注册

void Purchaser::Logging() {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

string user\_name;

string password1, password2;

cout<<endl<<" 请输入注册用户名: ";

cin >> user\_name;

//验证用户名是否存在

ifstream InMyFile1;

InMyFile1.open(PURCHASER\_FILE\_NAME);

string file\_user\_name, file\_password;

while (!InMyFile1.eof()) {

InMyFile1>>file\_user\_name>>file\_password;

if (file\_user\_name == user\_name) {

InMyFile1.close();

cout<<endl<<" 该用户名已被注册，注册失败"<<endl;

MY\_PAUSE();

return ;

}

}

InMyFile1.close();

cout<<endl<<" 请输入密码(不小于8位): ";

cin >> password1;

if (password1.length() < 8) {

cout<<endl<<" 密码长度应不小于8位，注册失败"<<endl;

MY\_PAUSE();

return ;

}

cout<<endl<<" 请再次输入密码: ";

cin >> password2;

MY\_FLUSH();

if (password1 != password2) {

cout<<endl<<" 两次输入的密码不一致，注册失败"<<endl;

MY\_PAUSE();

return ;

}

fstream InMyFile2;

InMyFile2.open(PURCHASER\_FILE\_NAME, ios::app);

InMyFile2.seekg(0, ios::end); //将文件指针指向文件末端

streampos fp = InMyFile2.tellg(); //fp为文件指针的偏移量

if (int(fp) != 0) // 偏移量为0，证明文件为空，为首次进入系统,不是首次进入系统就换行

InMyFile2<<endl;

InMyFile2<<user\_name<<" "<<password1;

InMyFile2.close();

cout <<endl<<" 注册成功"<<endl;

MY\_PAUSE();

}

//买家登录

Purchaser Purchaser::Loggin() {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

string user\_name;

string password;

cout<<endl<<" 请 输 入 用 户 名:";

cin >> user\_name;

cout<<endl<<" 请 输 入 密 码:";

cin >> password;

string file\_user\_name; //文件中的用户名

string file\_password; //文件中的密码

ifstream OutMyFile;

OutMyFile.open(PURCHASER\_FILE\_NAME);

while (!OutMyFile.eof()) {

OutMyFile>>file\_user\_name>>file\_password;

if (file\_user\_name == user\_name) {

if (file\_password == password) {

OutMyFile.close();

cout<<endl<<" 密码正确,正在登录...."<<endl;

sleep(2);

Purchaser p(user\_name, password);

p.ModifyLogStatus(true);

return p;

} else {

OutMyFile.close();

cout<<endl<<" 密码错误，正在返回买家界面...."<<endl;

sleep(2);

Purchaser p;

return p;

}

}

}

cout<<endl<<" 该用户名不存在，正在返回买家界面...."<<endl;

sleep(2);

OutMyFile.close();

Purchaser p;

return p;

}

//查看所有上架花卉

void Purchaser::CheckALLFlower() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

printf(" 花卉名称 花店名 花卉颜色 单价 数量 上架日期 \n");

for (int i = 0; i < MAXSIZE; ++i) {

LNode \*p = hashtable.first[i];

while (p != NULL) {

printf(" %-8s %-8s %-5s %5.1f %-3d ", p->flower->FlowerNameInfo().c\_str(), p->flower->ShopNameInfo().c\_str(), p->flower->PetalColorInfo().c\_str(),p->flower->FlowerPriceInfo(), p->flower->FlowerNumberInfo());

//用到运算符重载 <<

cout<<p->flower->FlowerDateInfo();

p = p->next;

usleep(25000);

}

}

}

//查找花卉

void Purchaser::SearchFlower() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

LNode\* linklist;

linklist = NULL;

string flower\_name;

cout <<endl<< " 请输入你要查找的花卉名称 :";

cin >> flower\_name;

int index = hashtable.SearchFlower(flower\_name, 1);

if (index == -1) {

cout <<endl<< " 你要查找的花卉不存在" << endl;

return ;

} else {

LNode \*p = hashtable.first[index];

cout <<endl<< " 你要查找的花卉信息为: " << endl;

printf(" 花卉名称 花店名 花卉颜色 单价 数量 上架日期 \n");

while (p != NULL) {

printf(" %-8s %-8s %-5s %5.1f %-3d ", p->flower->FlowerNameInfo().c\_str(), p->flower->ShopNameInfo().c\_str(), p->flower->PetalColorInfo().c\_str(),p->flower->FlowerPriceInfo(), p->flower->FlowerNumberInfo());

//用到运算符重载 <<

cout<<p->flower->FlowerDateInfo();

p = p->next;

usleep(25000);

}

bool flag = 1;

LNode \*q;

while (flag) {

cout <<endl<< " 1.按单价排序" << endl;

cout <<endl<< " 2.按数量排序" << endl;

cout <<endl<< " 3.按上架日期排序" << endl;

cout <<endl<< " 4.返回" << endl;

char ch;

cin >> ch;

fflush(stdin);

switch (ch) {

case '1':

system("clear");

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

q = MergeSort1(hashtable.first[index]);

hashtable.first[index] = q;

cout <<endl<< " 按单价排序: " << endl<<endl;

printf(" 花卉名称 花店名 花卉颜色 单价 数量 上架日期 \n");

while (q != NULL) {

printf(" %-8s %-8s %-5s %5.1f %-3d %-2d/%-2d/%-2d\n", q->flower->FlowerNameInfo().c\_str(), q->flower->ShopNameInfo().c\_str(), q->flower->PetalColorInfo().c\_str(),q->flower->FlowerPriceInfo(), q->flower->FlowerNumberInfo(),

q->flower->YearInfo(), q->flower->MonthInfo(), q->flower->DayInfo());

q = q->next;

usleep(25000);

}

MY\_PAUSE();

break;

case '2':

system("clear");

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

q = MergeSort2(hashtable.first[index]);

hashtable.first[index] = q;

cout <<endl<< " 按数量排序: " << endl <<endl;

printf(" 花卉名称 花店名 花卉颜色 单价 数量 上架日期 \n");

while (q != NULL) {

printf(" %-8s %-8s %-5s %5.1f %-3d %-2d/%-2d/%-2d\n", q->flower->FlowerNameInfo().c\_str(), q->flower->ShopNameInfo().c\_str(), q->flower->PetalColorInfo().c\_str(),q->flower->FlowerPriceInfo(), q->flower->FlowerNumberInfo(),

q->flower->YearInfo(), q->flower->MonthInfo(), q->flower->DayInfo());

q = q->next;

usleep(25000);

}

MY\_PAUSE();

break;

case '3':

system("clear");

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

q = MergeSort3(hashtable.first[index]);

hashtable.first[index] = q;

cout <<endl<< " 按上架日期排序: " << endl<<endl;

printf(" 花卉名称 花店名 花卉颜色 单价 数量 上架日期 \n");

while (q != NULL) {

printf(" %-8s %-8s %-5s %5.1f %-3d %-2d/%-2d/%-2d\n", q->flower->FlowerNameInfo().c\_str(), q->flower->ShopNameInfo().c\_str(), q->flower->PetalColorInfo().c\_str(),q->flower->FlowerPriceInfo(), q->flower->FlowerNumberInfo(),

q->flower->YearInfo(), q->flower->MonthInfo(), q->flower->DayInfo());

q = q->next;

usleep(25000);

}

MY\_PAUSE();

break;

case '4':

flag = 0;

break;

default:

cout <<endl<< " 无效的输入,请重新输入" << endl;

}

}

return ;

}

}

//价格归并排序1

LNode\* Purchaser::MergeList1(LNode \*L1, LNode \*L2) const {

LNode \*p, \*q, \*L, \*r;

p = L1;

q = L2;

if(p&&q&&p->flower->FlowerPriceInfo()<=q->flower->FlowerPriceInfo()){

L = p;

p = p->next;

r = L;

}

else if(p&&q){

L = q;

q = q->next;

r = L;

}

while(p&&q){

if(p->flower->FlowerPriceInfo()<=q->flower->FlowerPriceInfo()){

r->next = p;

r = p;

p = p->next;

}

else{

r->next = q;

r = q;

q = q->next;

}

}

p = p?p:q;

r->next = p;

return L;

}

//价格归并排序2

LNode\* Purchaser::MergeSort1(LNode \*head) const {

if(head==NULL||head->next==NULL)

return head;

LNode \*slow, \*fast;

slow = head;

fast = head->next;

while(fast!=NULL){

fast = fast->next;

if(fast!=NULL){

fast = fast->next;

slow = slow->next;

}

}

LNode \*lefthead = head, \*righthead = slow->next;

slow->next = NULL;

lefthead = MergeSort1(lefthead);

righthead = MergeSort1(righthead);

return MergeList1(lefthead, righthead);

}

//数量归并排序1

LNode\* Purchaser::MergeList2(LNode \*L1, LNode \*L2) const {

LNode \*p, \*q, \*L, \*r;

p = L1;

q = L2;

if(p&&q&&p->flower->FlowerNumberInfo()<=q->flower->FlowerNumberInfo()){

L = p;

p = p->next;

r = L;

}

else if(p&&q){

L = q;

q = q->next;

r = L;

}

while(p&&q){

if(p->flower->FlowerNumberInfo()<=q->flower->FlowerNumberInfo()){

r->next = p;

r = p;

p = p->next;

}

else{

r->next = q;

r = q;

q = q->next;

}

}

p = p?p:q;

r->next = p;

return L;

}

//数量归并排序2

LNode\* Purchaser::MergeSort2(LNode \*head) const {

if(head==NULL||head->next==NULL)

return head;

LNode \*slow, \*fast;

slow = head;

fast = head->next;

while(fast!=NULL){

fast = fast->next;

if(fast!=NULL){

fast = fast->next;

slow = slow->next;

}

}

LNode \*lefthead = head, \*righthead = slow->next;

slow->next = NULL;

lefthead = MergeSort2(lefthead);

righthead = MergeSort2(righthead);

return MergeList2(lefthead, righthead);

}

//上架日期归并排序1

LNode\* Purchaser::MergeList3(LNode \*L1, LNode \*L2) const {

LNode \*p, \*q, \*L, \*r;

p = L1;

q = L2;

if(p&&q&&p->flower->YearInfo()==q->flower->YearInfo()){

if(p->flower->MonthInfo()==q->flower->MonthInfo()){

if(p->flower->DayInfo()<=q->flower->DayInfo()){

L = p;

p = p->next;

r = L;

} else {

L = q;

q = q->next;

r = L;

}

} else if (p->flower->MonthInfo()<=q->flower->MonthInfo()) {

L = p;

p = p->next;

r = L;

} else {

L = q;

q = q->next;

r = L;

}

} else if (p&&q&&p->flower->YearInfo()<q->flower->YearInfo()) {

L = p;

p = p->next;

r = L;

} else {

L = q;

q = q->next;

r = L;

}

while(p&&q){

if(p->flower->YearInfo()<=q->flower->YearInfo()){

if(p->flower->MonthInfo()<=q->flower->MonthInfo()) {

if(p->flower->DayInfo()<=q->flower->DayInfo()) {

r->next = p;

r = p;

p = p->next;

} else {

r->next = q;

r = q;

q = q->next;

}

} else {

r->next = q;

r = q;

q = q->next;

}

} else {

r->next = q;

r = q;

q = q->next;

}

}

p = p?p:q;

r->next = p;

return L;

}

//上架日期归并排序2

LNode\* Purchaser::MergeSort3(LNode \*head) const {

if(head==NULL||head->next==NULL)

return head;

LNode \*slow, \*fast;

slow = head;

fast = head->next;

while(fast!=NULL){

fast = fast->next;

if(fast!=NULL){

fast = fast->next;

slow = slow->next;

}

}

LNode \*lefthead = head, \*righthead = slow->next;

slow->next = NULL;

lefthead = MergeSort3(lefthead);

righthead = MergeSort3(righthead);

return MergeList3(lefthead, righthead);

}

//预约花卉

void Purchaser::ReserveFlower() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

string flower\_name;

cout <<endl<< " 请输入您要预约的花卉名称 :";

cin >> flower\_name;

fflush(stdin);

int index = hashtable.SearchFlower(flower\_name, 1);

if (index == -1) {

cout <<endl<< " 您要预约的花卉不存在" << endl;

return ;

} else {

LNode \*p = hashtable.first[index];

cout <<endl<< " 您可以预约的花卉如下: " << endl;

printf(" 花卉名称 花店名 花卉颜色 单价 数量 上架日期 \n");

while (p != NULL) {

printf(" %-8s %-8s %-5s %5.1f %-3d %-2d/%-2d/%-2d\n", p->flower->FlowerNameInfo().c\_str(), p->flower->ShopNameInfo().c\_str(), p->flower->PetalColorInfo().c\_str(),p->flower->FlowerPriceInfo(), p->flower->FlowerNumberInfo(),

p->flower->YearInfo(), p->flower->MonthInfo(), p->flower->DayInfo());

p = p->next;

}

string shop\_name;

cout <<endl<< " 请输入您要预约花卉的花店名: ";

cin >> shop\_name;

fflush(stdin);

p = hashtable.first[index];

while (p != NULL) {

if (p->flower->ShopNameInfo() == shop\_name) {

cout <<endl<< " 请输入您要预约的数量: ";

int num;

cin >> num;

fflush(stdin);

if (p->flower->FlowerNumberInfo() < num) {

cout <<endl<< " 预约数量大于库存，预约失败" << endl;

return ;

} else {

p->flower->ModifyFlowerNumber(p->flower->FlowerNumberInfo() - num);

string file\_name = FILE\_ADDRESS + UserNameInfo() + FILE\_TYPE;

fstream InMyFile;

InMyFile.open(file\_name, ios::app);

InMyFile.seekg(0, ios::end); //将文件指针指向文件末端

streampos fp = InMyFile.tellg(); //fp为文件指针的偏移量

if (int(fp) != 0) // 偏移量为0，证明文件为空，为首次进入系统,不是首次进入系统就换行

InMyFile<<endl;

InMyFile<<p->flower->FlowerNameInfo()<<" "<<p->flower->ShopNameInfo()<<" "<<p->flower->PetalColorInfo()<<" "<<p->flower->FlowerPriceInfo()<<" "<<num<<" "<<p->flower->YearInfo()<<" "<<p->flower->MonthInfo()<<" "<<

p->flower->DayInfo();

InMyFile.close();

cout <<endl<< " 预约成功" << endl;

return ;

}

}

p = p->next;

}

cout <<endl<< " 该商家不存在" << endl;

return ;

}

}

//查看预约

void Purchaser::CheckReservation() const{

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

cout <<endl<< " 您预约的花卉如下: " <<endl<<endl;

string flower\_name;

string shop\_name;

string flower\_color;

float flower\_price;

int flower\_number;

int year, month, day;

fstream OutMyFile;

string file\_name = FILE\_ADDRESS + UserNameInfo() + FILE\_TYPE;

OutMyFile.open(file\_name);

OutMyFile.seekg(0, ios::end); //将文件指针指向文件末端

streampos fp = OutMyFile.tellg(); //fp为文件指针的偏移量

if (int(fp) == 0) { // 偏移量为0，证明文件为空，为首次进入系统,不是首次进入系统就换行

cout <<endl<< " 您没有预约花卉" << endl;

return ;

}

OutMyFile.close();

OutMyFile.open(file\_name);

printf(" 花卉名称 花店名 花卉颜色 单价 数量 上架日期 \n");

while (!OutMyFile.eof()) {

OutMyFile>>flower\_name>>shop\_name>>flower\_color>>flower\_price>>flower\_number>>year>>month>>day;

printf(" %-8s %-8s %-5s %5.1f %-3d %-2d/%-2d/%-2d\n", flower\_name.c\_str(), shop\_name.c\_str(), flower\_color.c\_str(),flower\_price, flower\_number,

year, month, day);

usleep(25000);

}

OutMyFile.close();

}

//花卉推荐

void Purchaser::RecommendFlower() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

// Sleep(5);

cout<<endl;

cout<<endl;cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" ");

cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

printf(" {@} {@} \n");

printf(" | ① 日 期 范 围 | \n");

printf(" \\|/ ② 鲜 花 用 途 \\|/ \n");

printf(" | | \n");

printf(" | | \n");

int n;

char ch, ch2;

cin >> ch;

fflush(stdin);

time\_t timep;

struct tm \*p;

time(&timep);

p=gmtime(&timep);

int flag = 0;

int now\_year, now\_month, now\_day;

now\_year = 1900+p->tm\_year;

now\_month = 1+p->tm\_mon;

now\_day = p->tm\_mday;

//圣诞节

const int shengdan\_year = now\_year, shengdan\_month = 12, shengdan\_day = 25;

switch (ch) {

case '1':

printf("\n 请设置日期范围:");

cin >> n;

fflush(stdin);

system("clear");

printf("\n");

cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" "); cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

printf("\n 今天是%d月%d日\n", now\_month, now\_day);

// Sleep(1);

if(DateCompare(now\_year, now\_month, now\_day, shengdan\_year, shengdan\_month, shengdan\_day)<=n){

printf("\n 距离圣诞节(12月25日)有%d天\n", DateCompare(now\_year, now\_month, now\_day, shengdan\_year, shengdan\_month, shengdan\_day));

// Sleep(1);

printf("送恋人:\n");

// Sleep(1);

printf("红玫瑰: 众所周知玫瑰是代表爱情的花朵，红玫瑰更是代表热情，爱恋的意思，要问给女朋友圣诞节送什么花?红玫瑰当然是首选了，既能表达你们的感情热情如火，又能增加浪漫度。\n");

// Sleep(1);

printf("百合: 在大家的心中百合花是纯洁无暇、天真无邪的，加上百合花的花语是纯洁的爱。在圣诞节这一天除了考虑送玫瑰花，也可以送百合花哦，如果是表白可以事先弄清楚对方的喜好，会事半功倍的。\n");

// Sleep(1);

printf("郁金香: 郁金香的花语为博爱、体贴、高雅、富贵、能干、聪颖。不同的花色也有着不同的花语，例如：红色郁金香代表热烈的爱意、粉色郁金香代表永远的爱、黄色郁金香代表开朗、白色郁金香代表纯洁清高的恋情、黑色郁金香代表爱的表白，永恒的祝福。\n");

// Sleep(1);

printf("送老师:\n");

// Sleep(1);

printf("康乃馨: 康乃馨的花语是爱，魅力和尊敬之情。像粉色康乃馨较适合送给女老师，黄色康乃馨较适合送给男老师。\n");

// Sleep(1);

printf("向日葵、石竹梅: 表达自己的感恩和祝福之情。\n");

// Sleep(1);

flag = 1;

}

if(flag==0){

printf("\n 您设置日期范围内没有节日\n");

printf("\n ");

}

printf("\n ");

break;

case '2':

system("clear");

printf("\n");

cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" "); cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

cout<<endl<<" 请选择事件: "<<endl;

cout<<endl<<" 1.爱情鲜花 "<<endl;

cout<<endl<<" 2.生日鲜花 "<<endl;

cout<<endl<<" 3.友情鲜花 "<<endl;

cout<<endl<<" 4.问候长辈 "<<endl;

cout<<endl<<" 5.探病慰问 "<<endl;

cout<<endl<<" 6.婚庆鲜花 "<<endl;

cout<<endl<<" 7.商务鲜花 "<<endl;

cin >> ch2;

fflush(stdin);

if(ch2=='1'){

system("clear");

printf("\n");

cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" "); cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

printf("\n 爱情鲜花:\n\n");

printf(" 郁金香:\n"

" 爱的表白、荣誉、祝福、永恒\n\n"

" 郁金香(紫):\n"

" 无尽的爱、最爱\n\n"

" 康乃馨(白):\n"

" 吾爱永在、真情、纯洁\n\n"

" 红玫瑰:\n"

" 热恋 粉玫瑰 永远的爱\n\n"

" 星辰花:\n"

" 永不变心\n\n"

" 孤挺花:\n"

" 喋喋不休\n\n"

" 嘉德利亚兰:\n"

" 贵妇人\n\n"

" 蝴蝶兰:\n"

" 幸福逐渐到来\n\n"

" 文心兰:\n"

" 隐藏的爱\n\n"

" 秋石斛:\n"

" 迷惑\n\n");

}

else if(ch2=='2'){

system("clear");

printf("\n");

cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" "); cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

printf("\n 生日鲜花\n\n");

printf(" 朋友生日:\n");

printf(" 红月季、满天星、石榴花、象牙红代表火红年华、前程似锦\n");

printf(" 长辈生日:\n");

printf(" 长寿花、白百合、万年青等,寓意福如东海，寿比南山\n");

printf(" 同事生日:\n");

printf(" 百合、天堂鸟、非洲菊、圣诞百合等,象征着祝福身体健康，事业蒸蒸日上\n");

printf(" 妈妈生日: \n");

printf(" 当然是送伟大的母爱之花——康乃馨了，粉色是女性的颜色，那康乃馨层层的花瓣代表着母亲对子女绵绵不断的感情。\n");

printf(" 送一束康乃馨给妈妈，是对妈妈生日的最大感恩。\n");

printf(" 夫妻生日: \n");

printf(" 百合象征着百年好合，长相厮守,玫瑰象征着爱意的表达\n");

}

else if(ch2=='3'){

system("clear");

printf("\n");

cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" "); cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

printf("\n 友情鲜花:\n\n");

printf(" 百合花：\n"

" 纯洁的友谊\n\n"

" 黄色鸢尾: \n"

" 友谊永固、热情开朗\n\n"

" 黄色夹竹桃：\n"

" 深刻的友情\n\n "

" 黄康乃馨：\n"

" 长久的友谊\n\n"

" 白康乃馨：\n"

" 纯结的友谊 \n\n"

" 勿忘我：\n"

" 浓情厚意，永恒的友谊 \n\n"

" 黄玫瑰花:\n"

" 友谊之情 \n\n"

" 木棉：\n"

" 友谊天长地久 \n\n"

" 百日草：\n"

" 想念远方朋友，天长地久\n\n"

" 万年春：\n"

" 友谊长存 \n\n"

" 秋海棠：\n"

" 诚挚的友谊\n \n"

" 茉莉花：\n"

" 纯洁、清操和友谊\n\n"

" 荷花：\n"

" 友谊的象征和使者\n\n");

}

else if(ch2=='4'){

system("clear");

printf("\n");

cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" "); cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

printf("\n 问候长辈\n\n");

printf(" 佛手：\n"

" 佛与福音近，被视为吉祥之物，赠送长者，寓意多福多寿\n\n"

" 菊花：\n"

" 别名“寿客”。为老人祝寿，颜色可挑红色、粉红色、紫色，寓意高风亮节，健康长寿\n\n"

" 长寿花：\n"

" 可用来祝福家乐长寿，吉祥\n\n"

" 富贵龟背竹：\n"

" 被称为“植物乌龟”象征长寿吉祥\n\n"

" 铁树：\n"

" 寓意吉祥\n\n"

" 长寿松树、鹤望兰：\n"

" 松鹤延年，是祝贺老人长寿的花木\n\n"

" 万年青、寿星草：\n"

" 祝贺老人延年益寿的花木\n\n"

" 牡丹：\n"

" 贺富贵荣华的鲜花\n\n"

" 剑兰：\n"

" 祝贺老人福寿康宁\n\n"

" 南天竹：\n"

" 祝贺幸福长寿的花木\n\n"

" 紫薇、一品红、仙客来：\n"

" 寓意健康长寿\n\n"

" 寿星鸡冠：\n"

" 祝老人幸福长寿\n\n");

}

else if(ch2=='5'){

system("clear");

printf("\n");

cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" "); cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

printf("\n 探病慰问\n\n");

printf(" 探病送花，含有关怀、慰问、祝福病患平安、早日康复之意。此时应选用花色、香味淡雅的鲜花，\n"

" 如唐菖蒲、兰花、金桔、六出花、玫瑰及康乃馨等。忌送白色、蓝色、黑色花卉，鉴于病人的心\n"

" 情极为复杂，探病送花要注意防止产生误会。尽可能送些病人平常所喜欢，或较为娇艳的花草，绝\n"

" 不可送白的、蓝的或黑的花卉。还要注意忌送的数目：4、9、13 。另外还可选择香石竹、月季花、\n"

" 水仙花、兰花等，配以文竹、满天星或石松，以祝愿贵体早日康复。\n");

printf(" 给病人送花有很多禁忌，探望病人时不要送整盆的花，以免病人误会为久病成根。香味很浓的花对\n"

" 手术病人不利，易引起咳嗽;颜色太浓艳的花，会刺激病人的神经，激发烦躁情绪;山茶花容易落蕾，\n"

" 被认为不吉利。看望病人宜送兰花、水仙、马蹄莲等，或选用病人平时喜欢的品种，有利病人怡情养性，\n"

" 早日康复。\n");

}

else if(ch2=='6'){

system("clear");

printf("\n");

cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" "); cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

printf("\n 婚庆鲜花:\n\n");

printf(" 玫瑰:\n"

" 玫瑰是爱情的最佳代言人，自然也是婚花的首选。但是不同颜色的玫瑰表达不同的祝福：红玫瑰表达“深情”,\n"

" 粉红玫瑰表达“爱的宣言”。当然，还有白玫瑰。婚礼花最好是鲜艳的颜色，所以不推荐使用白玫瑰。\n\n");

printf(" 蝴蝶兰: \n"

" 蝴蝶兰代表着情侣间的真爱。它看起来像一只色彩鲜艳的蝴蝶，花美丽动人。在婚礼上出现，绝对是一件非常\n"

" 漂亮的装饰品。\n\n");

printf(" 百合花: \n"

" 百合花象征着纯洁和纯真。百合花加着麦穗作为新娘的头饰，意味着一百年的丰收。在现代婚礼中，百合花也\n"

" 很受欢迎，因为它们具有“百年好合”的含义。\n\n");

printf(" 康乃馨: \n"

" 康乃馨代表着爱、魅力和尊重，作为婚花，可以表达“相敬如宾、”的祝福。\n\n");

printf(" 剑兰: \n"

" 剑兰代表怀旧之情，也表达爱、意、寿、康、福。它隐含着“幸福与美满”的含义。\n\n");

printf(" 涪陵花: \n"

" 涪陵花一般用来祝福“事业成功，前途光明”，但它也象征着坚持不懈，不怕困难。它能表达祝愿新婚夫妇相互\n"

" 尊重和相爱的美好意义。\n\n");

}

if(ch2=='7'){

system("clear");

printf("\n");

cout<<TT;cout<<TT;

printf(" "); cout<<STAR; printf(" "); cout<<SOFTWARE;printf(" ");cout<<STAR; cout<<NEW\_LINE; cout<<NEW\_LINE;

printf("\n 商务鲜花\n\n");

printf(" 送客户:\n:"

" 老虎百合【卷丹花】---财富，骄傲，繁荣\n\n"

" 喇叭水仙---尊敬\n\n"

" 天堂鸟---自由，幸福，吉祥\n\n"

" 金百合【女性】---高贵艳丽中显纯洁\n\n"

" 薄荷花---美德\n\n");

printf(" 送老板: \n");

printf(" 一般情况送领导可以送百合、康乃馨，向日葵，如果你是女孩子，领导是先生那么可以考虑送玫瑰（玫瑰色一\n"

" 定不能送红玫瑰，以免误会），反之男下属送女上司也是一样，女下属送女上司这个红玫瑰无关尽要，只要领\n"

" 导喜欢，为了更好的表达情感百合其实是不错的选择，全球共认的吉祥花，它的花朵大，花期长，并且还有香\n"

" 味，做出来的花束会比别的花更显得大气，可以送一大束百合。\n\n");

printf(" 送领导: \n");

printf(" 如果是女领导的话，送那种不要带有强烈感情色彩的花就行，红玫瑰一定不适合，送康乃馨+百合花或许比较好。\n"

" 如果是给男领导送花以盆栽花卉为宜。盆栽花卉气派、有生命力，不管是摆放在宽敞的办公室还是家里，都具有\n"

" 很好的装饰作用。像大花蕙兰、君子兰等。\n\n");

}

printf("\n ");

break;

default:

printf("\n 输入有误\n\n");

printf("\n ");

break;

}

}

//比较日期，返回差值 (yy2/mm2/dd2 - yy1/mm1/dd1)

const int Purchaser::DateCompare(int yy1, int mm1, int dd1, int yy2, int mm2, int dd2) const {

int sum = 0;

if (mm1==mm2) {

return dd2 - dd1;

}

else if (mm1 < mm2-1) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if ((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

while (mm1 < mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31;

break;

case 4:case 6:case 9:case 11:

sum += 30;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29;

else

sum += 28;

}

mm1++;

}

sum += dd2 - 1;

return sum;

}

else if (mm1 < mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

sum += dd2 - 1;

return sum;

}

else if (mm1 > mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

while (mm1 < 13) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31;

break;

case 4:case 6:case 9:case 11:

sum += 30;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29;

else

sum += 28;

}

mm1++;

}

mm1 = 1;

if(mm1 < mm2-1){

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if((yy2%4==0 && yy2%100!=0)||yy2%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

while(mm1 < mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31;

break;

case 4:case 6:case 9:case 11:

sum += 30;

break;

case 2:

if((yy2%4==0 && yy2%100!=0) || yy2%400==0)

sum += 29;

else

sum += 28;

}

sum += dd2 - 1;

return sum;

}

}

else if (mm1 < mm2) {

switch (mm1) {

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

sum += 31 - dd1 + 1;

break;

case 4:case 6:case 9:case 11:

sum += 30 - dd1 + 1;

break;

case 2:

if((yy1%4==0 && yy1%100!=0) || yy1%400==0)

sum += 29 - dd1 + 1;

else

sum += 28 - dd1 + 1;

}

sum += dd2 - 1;

return sum;

}

}

return 0;

}

//花店地址导航

void Purchaser::NavigateShopAddress() const {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

Map m;

m.InitMap();

cout <<endl<< " 请输入要搜索的商家店名: ";

string shop\_name;

cin >> shop\_name;

MapNode node = m.SearchMap(shop\_name);

if (node.x == -1) {

cout <<endl<< " 您要搜索的商家不存在" << endl;

return ;

}

cout <<endl<< " 地图信息如下: "<<endl<<endl;

m.ShowMap();

cout <<endl<< " 请输入您所在位置: ";

char x, y;

cin >> x >> y;

path\_node start = m.locate(x, y);

m.ChangeMap(node, start);

cout <<endl<< " 已为您规划最短路线: " << endl;

m.ShowMap();

cout << endl;

}

// Purchaser.hpp

#ifndef Purchaser\_hpp

#define Purchaser\_hpp

#include <iostream>

#include <string>

#include <fstream>

#include <cstdio>

#include <unistd.h>

#include "Global.hpp"

#include "Flower.hpp"

#include "Hash.hpp"

#include "User.hpp"

#include "Map.hpp"

//买家类,继承用户类

class Purchaser : virtual public User {

public:

Purchaser(); //默认构造函数

Purchaser(const string &user\_name, const string &password);

void Logging(); //买家注册，注册成功返回true，失败返回false

Purchaser Loggin(); //买家登录，登录成功返回true，失败返回false

void CheckALLFlower() const; //查看所有商家花卉

void SearchFlower() const; //查找花卉

void ReserveFlower() const; //花卉预约

void CheckReservation() const; //查看预约

void RecommendFlower() const; //花卉推荐

const int DateCompare(int yy1, int mm1, int dd1, int yy2, int mm2, int dd2) const; //比较日期返回差值(yy2/mm2/dd2 - yy1/mm1/dd1)

void NavigateShopAddress() const; //花店地址导航

LNode\* MergeList1(LNode \*L1, LNode \*L2) const; //价格归并排序

LNode\* MergeSort1(LNode \*head) const; //价格归并排序

LNode\* MergeList2(LNode \*L1, LNode \*L2) const; //数量归并排序

LNode\* MergeSort2(LNode \*head) const; //数量归并排序

LNode\* MergeList3(LNode \*L1, LNode \*L2) const; //日期归并排序

LNode\* MergeSort3(LNode \*head) const; //日期归并排序

private:

LNode \*LinkList; //存放预约的花卉的链表

};

#endif /\* Purchaser\_hpp \*/

// Admin.cpp

#include "Admin.hpp"

extern char leftad1[30], leftad2[30], leftad3[30], leftad4[30], leftad5[30], leftad6[30], leftad7[30], leftad8[30], leftad9[30], leftad10[30];

extern char rightad1[30], rightad2[30], rightad3[30], rightad4[30], rightad5[30], rightad6[30], rightad7[30], rightad8[30], rightad9[30], rightad10[30];

extern char announcement1[120], announcement2[120], announcement3[120], announcement4[120], announcement5[120];

//默认构造函数

Admin::Admin() { }

//管理员登录

bool Admin::Loggin() {

cout<<endl<<" 请输入管理员密码: ";

string pass\_word;

cin >> pass\_word;

fflush(stdin);

if (pass\_word == admin\_password) {

cout<<endl<<" 密码正确,正在登录...."<<endl;

sleep(2);

return true;

}

cout<<endl<<" 密码错误，正在返回主菜单...."<<endl;

sleep(2);

return false;

}

//发布公告

void Admin::Announce() {

ClearAn();

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

cout<<endl<<" 请输入公告内容: "<<endl;

string announcement;

cin >> announcement;

fflush(stdin);

ofstream InMyFile;

InMyFile.open(ANNOUNCEMENT\_FILE\_NAME);

InMyFile<<announcement;

InMyFile.close();

cout<<endl<<" 公告发布成功"<<endl;

}

//清空公告

void Admin::ClearAn() {

for (int i = 0; i < 120; i++)

announcement1[i] = '\0';

for (int i = 0; i < 120; i++)

announcement2[i] = '\0';

for (int i = 0; i < 120; i++)

announcement3[i] = '\0';

for (int i = 0; i < 120; i++)

announcement4[i] = '\0';

for (int i = 0; i < 120; i++)

announcement5[i] = '\0';

}

//广告设置

void Admin::SetAd() {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

cout<<endl<<" 您要设置左栏广告（1）还是右栏广告（2）: ";

char ch;

cin >> ch;

fflush(stdin);

if (ch == '1') {

ClearAd2('1');

cout<<endl<<" 请输入新的左栏广告(小于80字): "<<endl;

string ad;

cin >> ad;

fflush(stdin);

ofstream InMyFile;

InMyFile.open(LEFT\_AD\_FILE\_NAME);

InMyFile<<ad;

cout<<endl<<" 左栏广告设置成功"<<endl;

} else if (ch == '2'){

ClearAd2('2');

cout<<endl<<" 请输入新的右栏广告(小于80字): "<<endl;

string ad;

cin >> ad;

fflush(stdin);

ofstream InMyFile;

InMyFile.open(RIGHT\_AD\_FILE\_NAME);

InMyFile<<ad;

cout<<endl<<" 右栏广告设置成功"<<endl;

} else {

cout<<endl<<" 输入有误"<<endl;

}

}

//广告一键清空

void Admin::ClearAd() {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

cout<<endl<<" 您要清空左栏广告（1）还是右栏广告（2）: ";

char ch;

cin >> ch;

fflush(stdin);

if (ch == '1') {

ofstream InMyFile;

InMyFile.open(LEFT\_AD\_FILE\_NAME);

string ad = "　　　广　　　　　　　　　　　　　　　告　　　　　　　　　　　　　　　位　　　　　　　　　　　　　　　招　　　　　　　　　　　　　　　租　　　　　　　　　　　　";

InMyFile<<ad;

cout<<endl<<" 左栏广告清空成功"<<endl;

} else if (ch == '2'){

ofstream InMyFile;

InMyFile.open(RIGHT\_AD\_FILE\_NAME);

string ad = "　　　广　　　　　　　　　　　　　　　告　　　　　　　　　　　　　　　位　　　　　　　　　　　　　　　招　　　　　　　　　　　　　　　租　　　　　　　　　　　　";

InMyFile<<ad;

cout<<endl<<" 右栏广告清空成功"<<endl;

} else {

cout<<endl<<" 输入有误"<<endl;

}

}

//广告置空，设置新的广告前先将广告置空;

void Admin::ClearAd2(char ch) {

switch (ch) {

case '1':

for (int i = 0; i < 30; i++)

leftad1[i] = '\0';

for (int i = 0; i < 30; i++)

leftad2[i] = '\0';

for (int i = 0; i < 30; i++)

leftad3[i] = '\0';

for (int i = 0; i < 30; i++)

leftad4[i] = '\0';

for (int i = 0; i < 30; i++)

leftad5[i] = '\0';

for (int i = 0; i < 30; i++)

leftad6[i] = '\0';

for (int i = 0; i < 30; i++)

leftad7[i] = '\0';

for (int i = 0; i < 30; i++)

leftad8[i] = '\0';

for (int i = 0; i < 30; i++)

leftad9[i] = '\0';

for (int i = 0; i < 30; i++)

leftad10[i] = '\0';

break;

case '2':

for (int i = 0; i < 30; i++)

rightad1[i] = '\0';

for (int i = 0; i < 30; i++)

rightad2[i] = '\0';

for (int i = 0; i < 30; i++)

rightad3[i] = '\0';

for (int i = 0; i < 30; i++)

rightad4[i] = '\0';

for (int i = 0; i < 30; i++)

rightad5[i] = '\0';

for (int i = 0; i < 30; i++)

rightad6[i] = '\0';

for (int i = 0; i < 30; i++)

rightad7[i] = '\0';

for (int i = 0; i < 30; i++)

rightad8[i] = '\0';

for (int i = 0; i < 30; i++)

rightad9[i] = '\0';

for (int i = 0; i < 30; i++)

rightad10[i] = '\0';

break;

}

}

//买家找回密码

void Admin::PurchaserRetrievePassword() {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

cout<<endl<<" 请输入用户名:";

string user\_\_name;

cin >> user\_\_name;

fflush(stdin);

typedef struct node {

string username;

string password;

struct node \*next;

}Node;

Node \*L = new Node;

Node \*p = L;

bool flag = false; //判断是否找到该用户

ifstream OutMyFile;

OutMyFile.open(PURCHASER\_FILE\_NAME);

while (!OutMyFile.eof()) {

Node \*q = new Node;

q->next = NULL;

OutMyFile>>q->username>>q->password;

if (q->username == user\_\_name) {

string password1, password2;

cout<<endl<<" 请输入修改后的密码: ";

cin >> password1;

if (password1.length() < 8) {

cout<<endl<<" 密码长度小于8位，修改失败"<<endl;

OutMyFile.close();

return ;

}

cout<<endl<<" 请再次输入密码:";

cin >> password2;

fflush(stdin);

if (password1 == password2) {

flag = true;

q->password = password1;

cout<<endl<<" 密码修改成功"<<endl;

} else {

cout<<endl<<" 两次输入的密码不一致，修改失败"<<endl;

OutMyFile.close();

return ;

}

}

p->next = q;

p = q;

}

OutMyFile.close();

if (flag == false)

cout<<endl<<" 未找到该用户"<<endl;

ofstream InMyFile;

InMyFile.open(PURCHASER\_FILE\_NAME);

L = L->next;

while (L != NULL) {

Node \*p = L;

InMyFile<<p->username<<" "<<p->password;

if (L->next != NULL)

InMyFile<<endl;

L = L->next;

delete p;

}

InMyFile.close();

}

//商家找回密码

void Admin::MerchantRetrievePassword() {

cout<<endl<<TT<<TT;

cout<<" "<<STAR<<" "<<SOFTWARE<<" "<<STAR<<endl<<endl;

cout<<endl<<" 请输入用户名:";

string user\_\_name;

cin >> user\_\_name;

fflush(stdin);

typedef struct node {

string username;

string password;

string shopname;

struct node \*next;

}Node;

Node \*L = new Node;

Node \*p = L;

bool flag = false; //判断是否找到该用户

ifstream OutMyFile;

OutMyFile.open(MERCHANT\_FILE\_NAME);

while (!OutMyFile.eof()) {

Node \*q = new Node;

q->next = NULL;

OutMyFile>>q->username>>q->password>>q->shopname;

if (q->username == user\_\_name) {

string password1, password2;

cout<<endl<<" 请输入修改后的密码: ";

cin >> password1;

if (password1.length() < 8) {

cout<<endl<<" 密码长度小于8位，修改失败"<<endl;

OutMyFile.close();

return ;

}

cout<<endl<<" 请再次输入密码:";

cin >> password2;

fflush(stdin);

if (password1 == password2) {

flag = true;

q->password = password1;

cout<<endl<<" 密码修改成功"<<endl;

} else {

cout<<endl<<" 两次输入的密码不一致，修改失败"<<endl;

OutMyFile.close();

return ;

}

}

p->next = q;

p = q;

}

OutMyFile.close();

if (flag == false)

cout<<endl<<" 未找到该用户"<<endl;

ofstream InMyFile;

InMyFile.open(MERCHANT\_FILE\_NAME);

L = L->next;

while (L != NULL) {

Node \*p = L;

InMyFile<<p->username<<" "<<p->password<<" "<<p->shopname;

if (L->next != NULL)

InMyFile<<endl;

L = L->next;

delete p;

}

InMyFile.close();

}

// Admin.hpp

#ifndef Admin\_hpp

#define Admin\_hpp

#include <iostream>

#include <string>

#include <fstream>

#include "Global.hpp"

#include "Flower.hpp"

#include "Hash.hpp"

#include "User.hpp"

#include "Merchant.hpp"

#include "Purchaser.hpp"

//管理员类,可进行任何操作,方便对系统进行测试，继承买家、卖家类，此时买家卖家类要设置虚基类

class Admin : public Merchant, public Purchaser {

public:

Admin();

bool Loggin(); //管理员登录，登录成功返回true，失败返回false

void Announce(); //发布公告

void ClearAn(); //清空公告

void SetAd(); //设置广告

void ClearAd(); //广告一键清空并显示: 广告位招租

void ClearAd2(char ch); //广告置空，设置新的广告前先将广告置空;

void MerchantRetrievePassword(); //商家找回密码

void PurchaserRetrievePassword(); //买家找回密码

private:

const string admin\_password = ADMIN\_PASSERWORD; //管理员密码

};

#endif /\* Admin\_hpp \*/