華中科技大學 人工智能与自动化学院
SCHOOL OF ARTIFICIAL INTELLIGENCE AND AUTOMATION.HUST

C 语言课程设计

校园外卖快递投递取送模拟系统

课程设计报告

专业班级：人工智能 2402 班

小组成员：曹瀚鹏 U202414188 张子恒 U202414212

指导老师：何顶新 左峥嵘 周纯杰 高常鑫

周凯波 汪国有 彭刚 陈忠

一、前言

1. 选题介绍

设计一个校园外卖快递的投递与去送模拟系统，包括**线上点餐**、**线上超市购物**、**快递代取**等功能。

2. 编写背景

随着移动互联网的快速发展，线上点餐、超市购物和快递代取等便捷服务已成为人们日常生活的重要组成部分。在校园环境中，由于学生和教职工的日常事务繁忙，对高效、便捷的外卖、超市配送及快递代取服务的需求日益增长。然而，当前校园内的配送服务仍存在诸多不足，例如食堂与超市缺乏系统化的线上订购渠道、配送效率较低、订单状态不可追踪、骑手路线规划不合理等问题。

因此，设计一款适用于校园场景的外卖快递投递取送模拟系统，将有助于优化校园配送服务，提高整体运作效率，提升用户体验。

3. 参考资料

《程序设计教程 用 C/C++语言编程》周纯杰 何顶新 周凯波 彭刚 等

北京：机械工业出版社 2016 年

二、任务概述

1. 目标用户分析

点餐及下单用户（学生及老师）：主要用于线上点餐、超市购物、快递代取服务。

商家（食堂餐饮经营者、超市商家）：提供餐饮和超市商品，并管理订单。

骑手（配送人员）：负责餐饮订单、超市订单及快递的配送。

2. 需求分析

本系统模拟了校园内外卖配送、超市购物配送及快递代取的完整流程，涵盖用户注册与管理、商家管理、订单处理、骑手配送等功能。通过合理的数据选取与系统设计，该系统可为校园内的学生、教师、食堂及超市商家、配送骑手等提供一体化的线上交易与配送服务。同时，结合骑手路径优化功能，该系统能够提高配送效率，减少配送成本，保障订单的及时送达，提升校园生活的便利性。

3. 数据选取及其处理

为了兼顾实用性和可行性，结合校园实际情况及编译环境的内存限制，本系统的数据选取如下：

外卖服务：拟将学校的 17 个食堂 作为提供外卖服务的商家。

超市购物服务：拟选取 3 家喻园学生超市 作为提供超市购物服务的商家。

快递代取：骑手配送的目的地设为 韵苑、紫菘、沁苑 南区 西区 五大社区。

骑手工作范围：基于 华中科技大学主校区的地图 作为骑手工作的场所，以合理的路线规划实现配送。

三、运行环境和配置

开发软件工具：Borland C++ 3.1

文字编辑工具：Visual Studio Code

操作系统：DOS Windows XP/10/11

四、主要功能

1. 实现账户的注册/登录

保存信息

2. 登陆后

根据账号的不同类型（用户，商家，骑手）显示相应账户端页面

3. 用户端

用户在首页可以输入自己的手机号，选择住址，并以区域+楼栋的模式存入文件，如果未录入全部信息，在下单时会弹出补充信息的窗口，将信息录入完整后方可下单

1）线上购物

用户点击“超市”按钮后可进入线上购物页面，页面中显示出不同商品的价格，名称，和在购物车中的数量，可以在页面顶部选择不同类型的商品，点击“+”号或“-”号可以更改购物车中商品的数量，当数量为0时若用户点击减号则会打印出提醒信息，数量不会成为负数。此外，页面上还提供了一个对商品进行排序的按钮，用户可以选择将商品按价格从高到低（从低到高）排序。

当用户点击“购物车”按钮时会进入购物车页面，页面中显示当前购物车内的商品数量价格以及总价，也可以在购物车中进行商品数量的增减处理。

当用户点击“生成订单“按钮时会进入订单页面，页面中将会显示当前订单的订单号，下单时间，用户名，用户手机号，用户地址以及的商品信息。当点击“确认并支付”按钮时订单信息会被存入到文件中并且在商家端和骑手端同步显示。如果信息未完善，下单时会提醒用户完善信息。

2）外卖点单

与线上购物大致相同，先选择所要下单的食堂，然后选择商品，最后确认并存入文件，同样在商家端和骑手端可以同步显示。

3）快递代取

用户输入取件码，选择所要取的快递服务商信息和所在驿站，然后进行下单，如果用户或快递信息未完善则会弹出提醒，当所有信息均完善后方可下单

4. 商家端

商家首次登录时需输入绑定码验证身份，然后选择所经营的食堂/超市（每个账号只可绑定一次，不可更改）。当绑定完成后，点击“查看订单”按钮“，就可以查看到当前经营的食堂/超市里的所有订单。首先显示简略信息（包括订单号，下单时间，下单用户，总金额），当点击订单时，会进入显示该订单详细信息的页面，并且可以开始备货，备货完成后将提醒骑手到店取单。

5. 骑手端

骑手在首次登录时需输入手机号绑定方可进行接单。信息录入后点击“接单”按钮可以看到当前的所有订单（线上超市购物，外卖，快递代取），并选择接取。选择后将自动进行路径规划，

五、数据处理及算法设计

1. 数据处理

1）骑手端地图来源：企业微信—校园地图—2D 地图（2023 版）

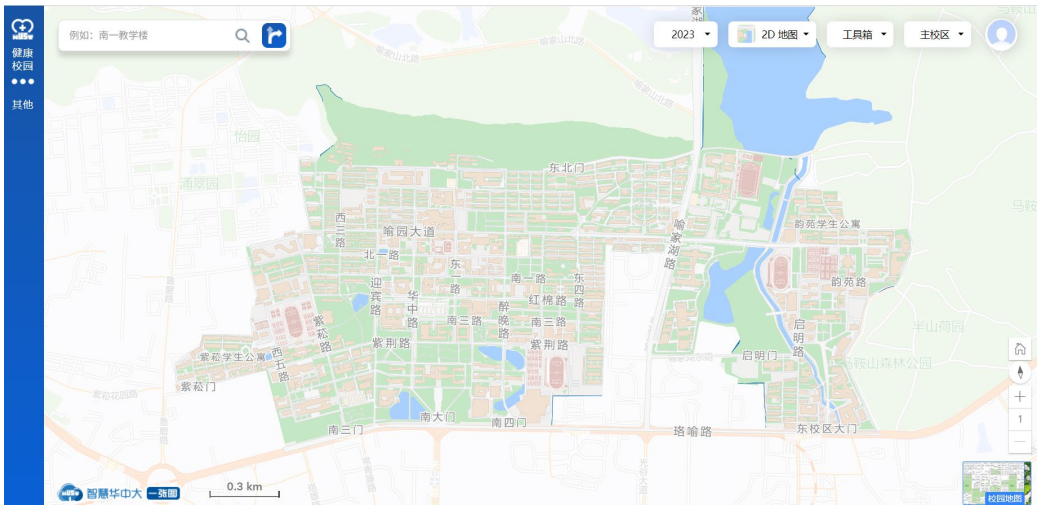
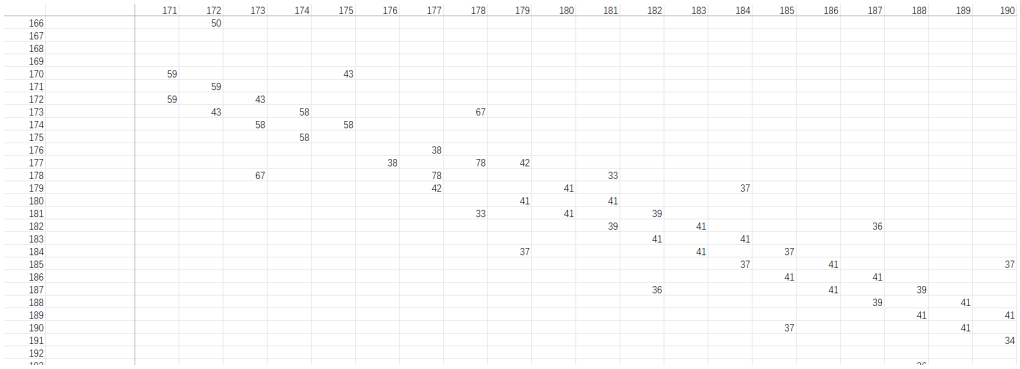


图 1 校园地图

2）节点设置：基于华中科技大学地图，共设置 417 个节点，涵盖全部主干道及主要分岔口，将图片压缩为 1024*442 尺寸后，在画图软件中测量每个节点的像素坐标，使用 excel 表格统计；在企业微信—校园地图—2D 地图（2023 版）--工具箱—距离量算获得任意两节点距离

3）数据统计：使用 excel 表格工具进行节点的位置坐标及距离的统计



	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190
166		50																		
167																				
168																				
169																				
170	59				43															
171		59																		
172	59		43																	
173		43		58				67												
174			58		58															
175				58																
176							38													
177						38		78	42											
178			67				78				33									
179							42			41				37						
180									41		41									
181								33		41		39								
182											39		41				36			
183												41		41						
184									37				41		37					
185														37		41				37
186															41		41			
187												36				41		39		
188															39				41	
189																		41		41
190															37				41	
191																				34
192																				

图 2 节点距离

序号	名称	坐标
1	韵苑学生食堂	854,143
2	东园学生食堂	971,165
3	学一食堂	857,239
4	学二食堂	857,228
5	东教工食堂	880,383
6	喻园食堂	381,130
7	集贤楼食堂	451,177
8	东一食堂	521,278
9	紫荆园餐厅	534,284
10	东三食堂	549,284
11	东四食堂	550,269
12	西一学生食堂	125,222
13	西二学生食堂	151,214
14	西三学生食堂	120,204
15	百景园餐厅	188,152
16	集锦园餐厅	380,112
17	百惠园餐厅	114,281
18	韵苑学生超市	858,143
19	喻园学生超市	713,139
20	西区学生超市	178,189
21	紫菘10栋	53,283
22	紫菘6栋	81,284
23	紫菘2栋	103,286
24	紫菘14栋	39,292
25	紫菘9栋	54,296
26	紫菘5栋	81,296
27	紫菘13栋	33,301

381	101,296
382	101,312
383	112,310
384	382,137
385	135,220
386	131,202
387	175,158
388	183,188
389	100,307
390	155,222
391	192,177
392	561,288
393	919,113
394	917,103
395	909，93
396	890,100
397	904,103
398	876,100
399	267,208
400	233,384
401	231,204
402	201,185
403	44,313
404	49,305
405	97,323
406	969,158
407	252,222
408	249,207

图 3 节点坐标

4）数据转化：使用 python 编写脚本，将节点的坐标，邻接节点索引，邻接节点数量，邻接节点间距离，节点名称等信息从 excel 表格中提取，导出为便于作为

结构体数组初始化的 txt 文本格式

```python
import pandas as pd
df = pd.read_excel("地图数据.xlsx", header=None)  # 若没有表头则设置 header=None
data = df.values.tolist()[1:]
result = []
for i in data:
    if ", " in str(i[2]):
        result.append(str(i[2]).split(', '))
    else:
        result.append([str(i[2])[:-3], str(i[2])[-3:]])
for i in result:
    i.append([])
    i.append([])
    i.append(0)
# 读取 Excel 文件（假设第一行为列名）
df = pd.read_excel("地图距离.xlsx", header=None)  # 若没有表头则设置 header=None
data = df.values.tolist()[1:]
result_index = 0
for i in data:
    i = i[2:]
    cnt = 1
    for j in i:
        if (not pd.isna(j)):
            result[result_index][2].append(cnt)
```

图 5 python-源代码 1

```python
            result[result_index][2].append(cnt)
            result[result_index][3].append(int(j))
            result[result_index][4] += 1
        cnt += 1
    result_index += 1
for i in result:
    while len(i[2]) < 6:
        i[2].append(0)
    while len(i[3]) < 6:
        i[3].append(20000)
    print(i)
# Node node = {0, 0, {0}, {0}, 0};
with open('result.txt', 'w', encoding='utf-8') as file:
    cnt = 1
    for row in result:
        file.write(rf'{{{row[0]}, {row[1]}, {{{row[2][0]}, {row[2][1]}, {row[2][2]}, {row[2][3]}, {row[2][4]}, '
                   rf'{row[2][5]}}}, {{{row[3][0]}, {row[3][1]}, {row[3][2]}, {row[3][3]}, {row[3][4]}, {row[3][5]}}},
        {row[4]}, "{cnt}号路口"}};' + '\n')
        cnt += 1
```

图 4 python-源代码 2

```
{854, 143, {131, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "1号路口"};
{971, 165, {406, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "2号路口"};
{857, 239, {366, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "3号路口"};
{857, 228, {365, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "4号路口"};
{880, 383, {153, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "5号路口"};
{381, 130, {384, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "6号路口"};
{451, 177, {246, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "7号路口"};
{521, 278, {197, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "8号路口"};
{534, 284, {195, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "9号路口"};
{549, 284, {194, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "10号路口"};
{550, 269, {202, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "11号路口"};
{125, 222, {385, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "12号路口"};
{151, 214, {385, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "13号路口"};
{120, 204, {386, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "14号路口"};
{188, 152, {387, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "15号路口"};
{380, 112, {254, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "16号路口"};
{114, 281, {329, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "17号路口"};
{858, 143, {130, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "18号路口"};
{713, 139, {154, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "19号路口"};
{178, 189, {388, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "20号路口"};
```

图 6 输出的 txt 格式节点数据

5）数据呈现：考虑到校园地图的节点数较多，但节点间连接较少，为稀疏图，故使用邻接表代替邻接矩阵的方式来呈现节点数据，以降低算法的时间复杂度

2. 算法设计：

1）两点间距离：结合编译器的运行性能，考虑图中节点和道路的数量，选用 dijkstra 算法计算任意两点间的最短路径，并将所遍历的节点逐一连线，形成可视化路线

2）骑手统筹安排：综合考虑数据量和界面尺寸，我们将骑手能接取的最大订单数设置为 4，对于每一个订

单，骑手都须先取餐后送餐，考虑到在地图上选择起点交互的困难，我们随机设置每次骑手的初始位置，而后计算骑手当前位置与其当前状态能到达节点（若改订单已取餐，则可对改订单进行配送；若为取餐，则只能去取餐点对于的节点）的距离，考虑到学校的地图多为横平竖直的道路，我们此时计算的是两点间的曼哈顿距离而非实际距离来作为统筹骑手路线的依据，能尽量平衡算法复杂度和准确性，使用贪心策略每次选择离自己最优的路径。统筹好路线后，使用 dijkstra 逐一计算当前位置到下一个任务点的最短路径，显示在地图中，并依据距离估算出大致时间供骑手参考（取校园内电动车 20km\h 计算时间），最终效果呈现良好

六、程序设计

1. 模块分析

1）基础模块：

a）鼠标文件：mouse.c

b）图形文件：SVGA.c　shape.c

c）文字显示文件：HZK.c

2）数据处理模块：

a) 订单信息存储以及显示：u_order.c　f_order.c

de_order.c　bu_order.c　bu_det.c

acp_or.c　acp_det.c　m_ac_de.c

m_acp.c　m_hst.c　m_inf.c

b) 路径规划：

route.c　arrange.c

c) 基本图形界面：

welcome.c　user.c　busi.c

rider.c　u_shop.c　u_food.c　u_deliv.c

u_cart.c

2. 核心程序流程图

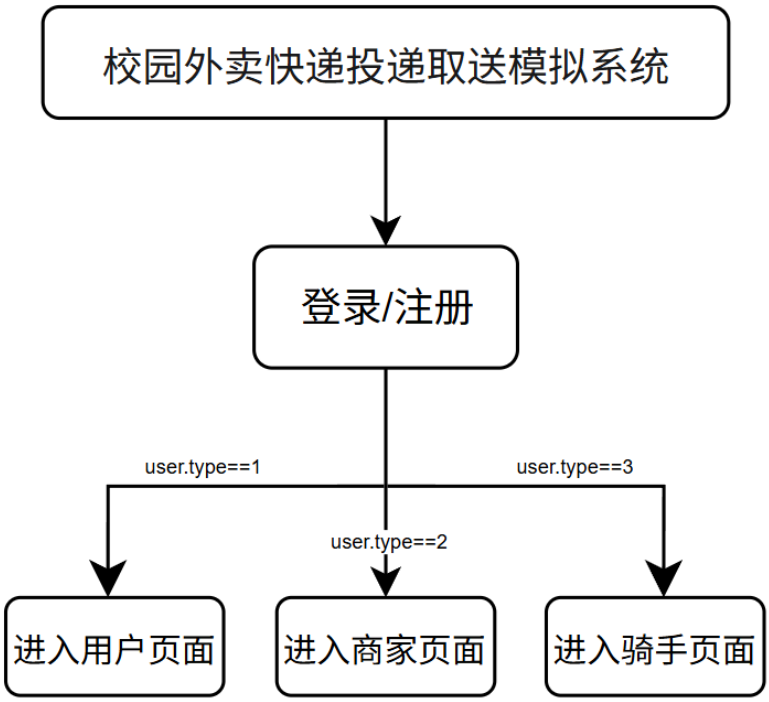

校园外卖快递投递取送模拟系统
登录/注册
user.type==1
user.type==2
user.type==3
进入用户页面
进入商家页面
进入骑手页面

图 7 主流程

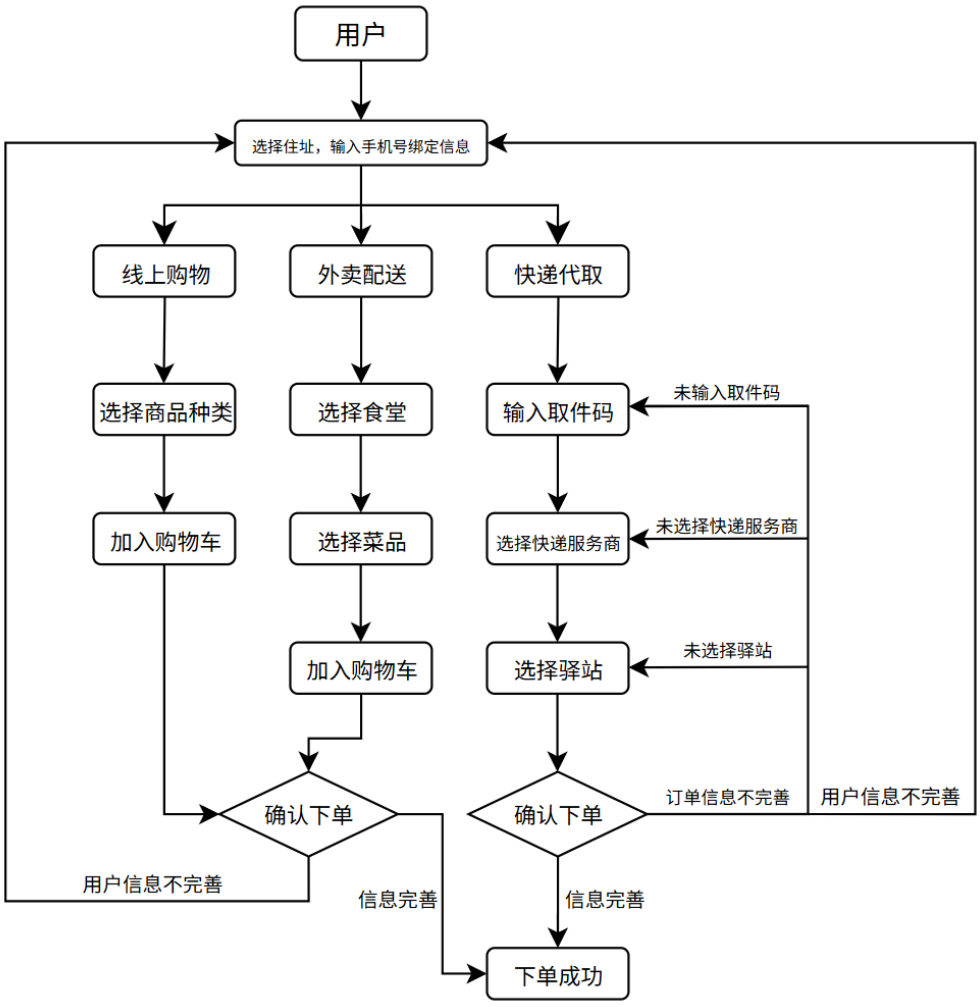

用户
选择住址，输入手机号绑定信息
线上购物
外卖配送
快递代取
选择商品种类
选择食堂
输入取件码
未输入取件码
加入购物车
选择菜品
选择快递服务商
未选择快递服务商
加入购物车
选择驿站
未选择驿站
确认下单
确认下单
订单信息不完善
用户信息不完善
用户信息不完善
信息完善
信息完善
下单成功

图 8 用户端

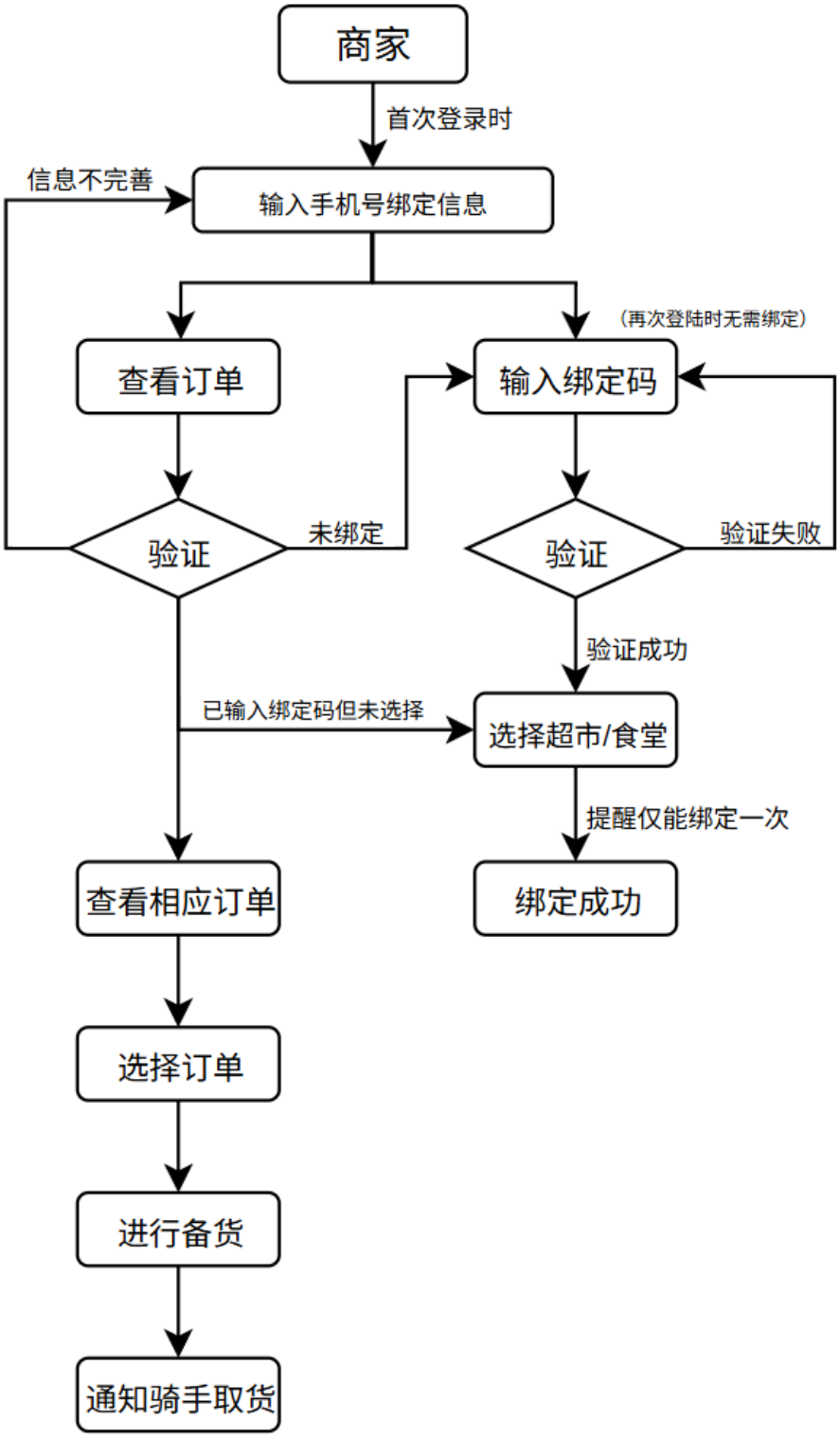

商家
首次登录时
信息不完善
输入手机号绑定信息
（再次登陆时无需绑定）
查看订单
输入绑定码
验证
未绑定
验证
验证失败
验证成功
已输入绑定码但未选择
选择超市/食堂
提醒仅能绑定一次
查看相应订单
绑定成功
选择订单
进行备货
通知骑手取货

图 9 商家端

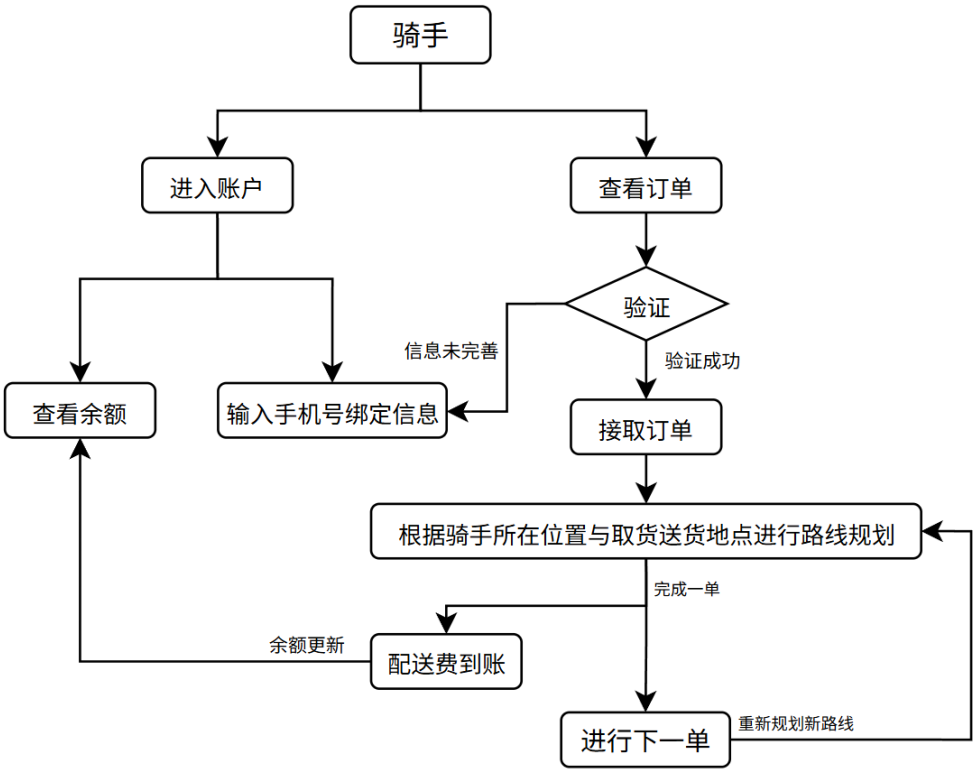


图 10 骑手端

3. 相关程序界面

图 11 登录界面

图 12 注册界面

图 13 用户端主界面

图 14 超市界面

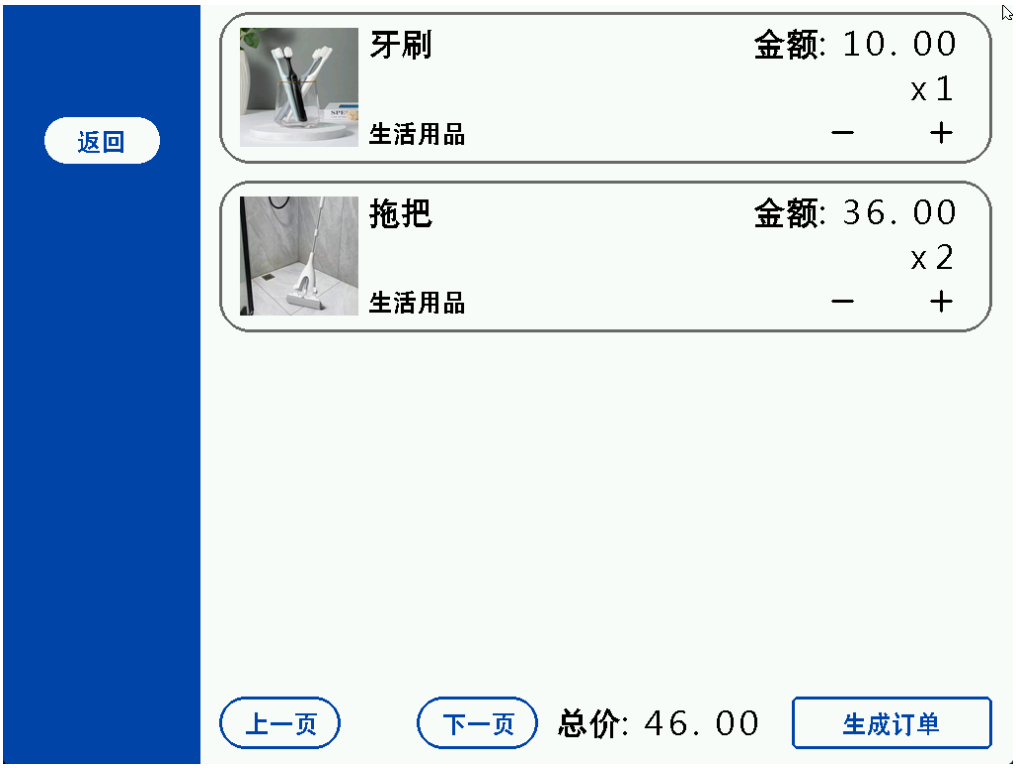


图 15 购物车界面

图 16 完善信息界面

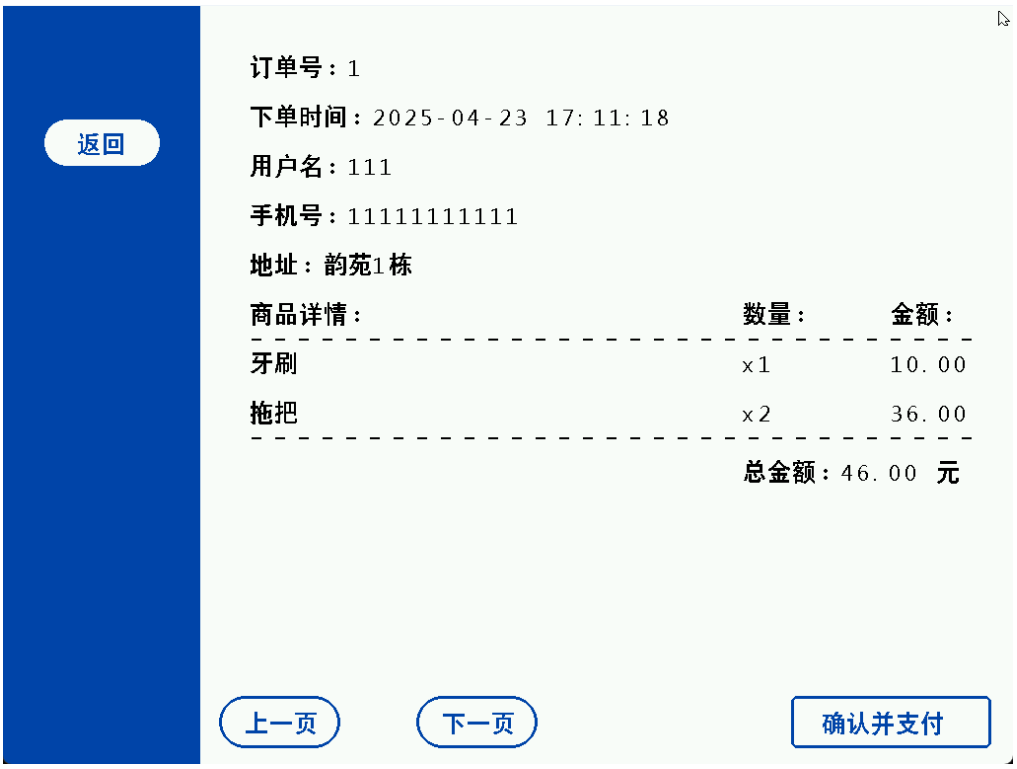


图 17 超市订单界面

图 18 商家端主界面

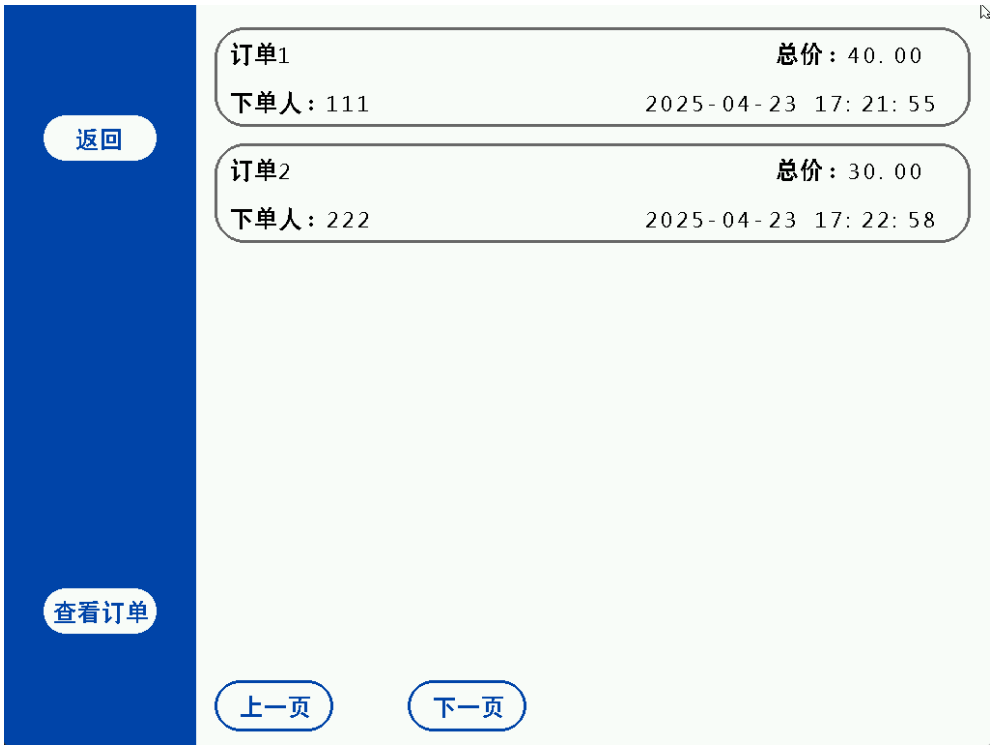


图 19 商家端查看订单界面

图 20 商家端查看订单界面

图 21 骑手绑定页面

图 22 骑手接单页面

图 23 骑手欢迎页面

图 24 骑手接单页面

图 25 骑手当前订单页面

图 26 骑手暂无订单页面

图 27 路线页面

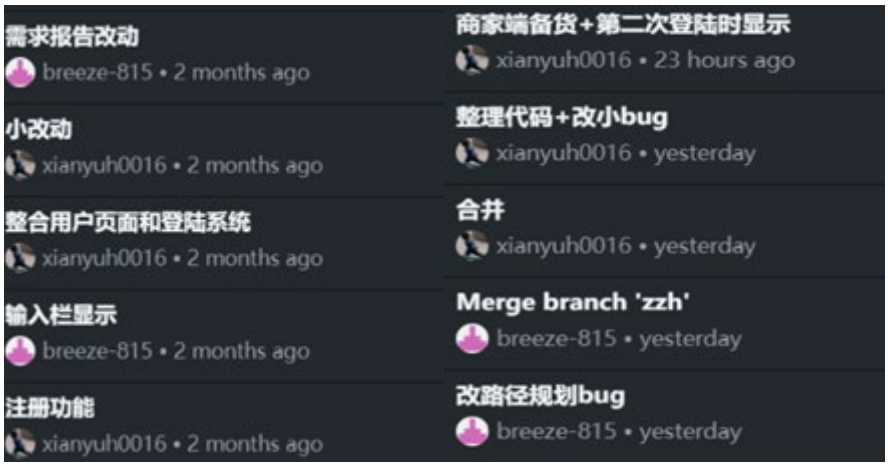
图 28 骑手暂无任务页面

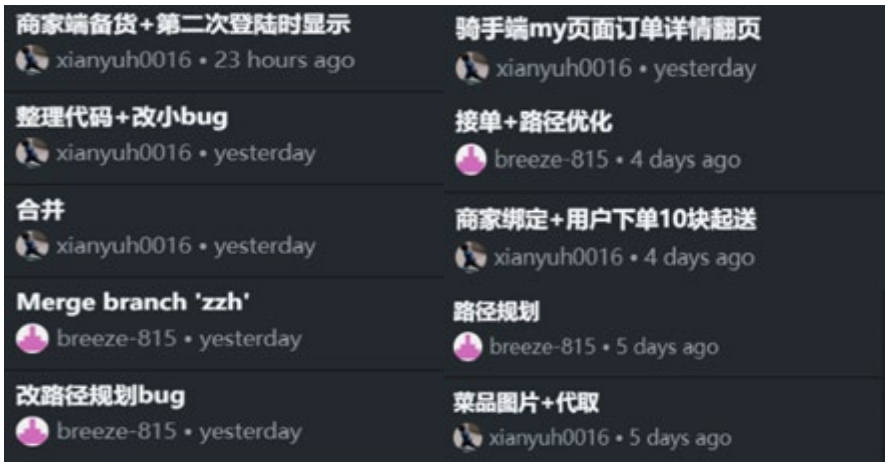
图 29 骑手历史订单页面

图 30 骑手信息页面

七、代码管理及工作日志

使用 GitHub desktop 实现协作及管理代码，创建两人合作仓库，使得每个版本更改的代码得以留存，同时大大提高两人合并代码的效率





八、源代码以及相关文件

1. data 文件

USERINFO.DAT（存储用户信息的文件）

ORDER.DAT（存储超市订单的文件）

FOODORDER.DAT（存储食堂订单的文件）

DELIVER.DAT（存储快递代取订单的文件）

2. 数据结构

```
typedef struct USER
{
    char name[12];//账户
    char code[12];//密码
    char type;//用户类型 1 为用户，2 为商家，3 为骑手
    char number[12];//手机号
    int community; //地址 1 东区 2 西区 3 南区 4 紫菘 5 韵苑
    int building;
    int state;//判断商家是否绑定，index，代表超市/食堂编号,未绑定为-1
    int pos;//用户在列表中位置
    int index; //用户住址索引
}USER;

typedef struct Product
{
    int id;
    int type;//种类
    char name[20];
    float price;
    char photo[50];
    int quantity;//数量
```

```
} Product;//商品

typedef struct CartItem{
    int id;//商品 id
    int type;//种类
    char name[20];//商品名称
    float price;//价格
    char photo[50];
    int quantity;//数量
    int index_in_products;//在商品数组中的索引，即 cnt
} CartItem;//购物车内的商品

typedef struct ShoppingCart{
    CartItem* items;//购物车内商品
    int itemCount;//购物车内商品种类数量
} ShoppingCart;//购物车整体

typedef struct Deliver
{
    int id; // 订单号
    int type; // 快递类型
    char name[10]; // 用户名
    char number[12];//手机号
    int community; //地址 1 东区 2 西区 3 南区 4 紫菘 5 韵苑
    int building;//楼栋号
    int destination;//送达地址
    char time[20]; // 下单时间
    char code[10]; //取件码
    int company; // 快递公司
    int station; // 站点
    int index;
    int total_cnt;
```

```
    int acp_count;
} Deliver;

typedef struct Food {
    int id; // 食物 ID
    int station; // 食堂编号
    char name[50]; // 食物名称
    float price; // 食物价格
    char photo[50]; // 食物图片路径
    int quantity; // 食物数量
} Food;

typedef struct FoodCart {
    int id;//商品 id
    int type;//种类
    char name[20];//商品名称
    float price;//价格
    char photo[50];
    int quantity;//数量
    int index_in_foods;//在商品数组中的索引，即 cnt
} FoodCart;//购物车内的商品

typedef struct ShoppingFood{
    FoodCart* items;//购物车内商品
    int itemCount;//购物车内商品种类数量
    int capacity;//购物车容量
} ShoppingFood;//购物车整体

typedef struct Order{
    int id;//订单号
    int community;//用户地址
    int building;//楼栋号
    int pick_up_location;//取餐地点
```

```
        int destination;
        char user_name[12];//用户名
        char user_phone[12];//用户手机号
        char order_time[20];//下单时间
        CartItem item[20];//购物车内物品信息
        int itemCount;//购物车内物品种类数量
        float total_amount;//总价
    } Order;//订单信息

    typedef struct AcceptedOrder{
        int type; // 0:超市, 1:食品, 2:快递
        union {
            Order order;
            FoodOrder food;
            Deliver deliver;
        } data;
    } AcceptedOrder;//已接取的订单
```

3. 头文件

```
    #ifndef __ACCOUNT_H__
    #define __ACCOUNT_H__

    void account();
    void draw_account();

    #endif
/*****************************************************************/
    #ifndef __ACP_DET_H
    #define __ACP_DET_H

    void draw_order_detail_header(int type, int local_index,OrderList
*OL, FoodList *FL, DeliverList *DL);
```

```
    void  draw_order_detail(int  type,OrderList  *OL,  FoodList  *FL,
DeliverList *DL,
        int local_index, int page);
    void show_order_detail(int local_index, int type, int user_pos) ;
    #endif
/*****************************************************************/
    #ifndef _PRINTCC_H_
    #define _PRINTCC_H_

    #define FANGSONG 0
    #define HEI 1
    #define KAI 2
    #define SONG 3
    #define SONG_BOLD 4
    #define TRADITIONAL 5
    #define FANGSONG_VERT 6
    #define HEI_VERT 7
    #define KAI_VERT 8
    #define SONG_VERT 9
    #define SONG_BOLD_VERT 10
    #define TRADITIONAL_VERT 11

    void  PrintCC(int  x0,int  y0,unsigned  char  str[],int  font,int
    size,int tracking,int color);
    void  PrintText(int  x0,int  y0,unsigned  char  str[],int  font,int
    size,int tracking,int color);

    #endif

/*****************************************************************/

    #ifndef ACP_ORDER_H
    #define ACP_ORDER_H
```

```
    void accept_order(int user_pos);
    double rider_deliver_price(int distance_m, float order_amount);
    void draw_accept_order(int page, OrderList *OL, FoodList *FL,
DeliverList *DL);
    void get_ordtyp_locind(int global_index,
        int *type, int *local_index,
         const OrderList *OL, const FoodList *FL, const DeliverList
*DL);
    #endif
/*****************************************************************/
    #ifndef __ALL_FUNC_H__
    #define __ALL_FUNC_H__

    #include<conio.h>
    #include<graphics.h>
    #include<stdio.h>
    #include<stdlib.h>
    #include<bios.h>
    #include<string.h>
    #include<dos.h>
    #include<math.h>
    #include<time.h>

    #include "mouse.h"
    #include "SVGA.H"

    #include "u_re.h"
    #include "welcome.h"
    #include "lgfunc.h"

    #include "user.h"
    #include "u_shop.h"
    #include "u_take.h"
```

```
#include "u_deliv.h"
#include "u_cart.h"
#include "u_order.h"
#include "u_food.h"
#include "f_order.h"
#include "button.h"
#include "de_order.h"

#include "busi.h"
#include "bu_order.h"
#include "bu_det.h"

#include "ride.h"
#include "route.h"
#include "acp_or.h"
#include "account.h"
#include "HZK.h"
#include "shape.h"
#include "arrange.h"
#include "acp_det.h"
#include "my_acp.h"

extern MOUSE mouse;

extern USER users;

extern Product products[84];
extern CartItem carts[84];
extern ShoppingCart cart;

extern Deliver delivers;

extern Food foods[12];
```

```
    extern FoodCart food_carts[12];
    extern ShoppingFood shopping_food;

    extern Order orders;
    extern Canteen canteen[17];

    extern Node node[417];
    extern Company companys[8];
    extern Station stations[8];

    extern Button button [79];
    extern AcceptedOrder acp_orders[4];

    extern RouteState route_state;
    #define white 0xFFFF
    #define snow 0xFFDF
    #define black 0x0000
    #define deepblue 0x0235
    #define lightblue 0x435c
    #define skyblue 0xB71C
    #define grey 0xC618
    #define lightred 0XF800
    #define Red 0xF800
    #define deepgrew 0XC618
    #define lightgrew 0xDEFB
    #endif
/*****************************************************************/
    #ifndef MOUSE_H
    #define MOUSE_H

    typedef struct {
        int x;
        int y;
```

```
    int key;
} MOUSE;

void mouse_init();
void MouseSpeed(int x, int y);
void MouseRange(int x1, int y1, int x2, int y2);
int MouseBut(MOUSE *mouse);
int MouseGet(MOUSE *mouse);
void MouseSet(int x, int y);
int mouse_press(int x1, int y1, int x2, int y2);

void mouse_on_arrow(MOUSE mouse);
void mouse_on_cursor(MOUSE mouse);
void mouse_on_hand(MOUSE mouse);

void mouse_show_cursor(MOUSE *mouse);
void mouse_show_arrow(MOUSE *mouse);
void mouse_show_hand(MOUSE *mouse);

void mouse_off_arrow(MOUSE *mouse);
void mouse_off_cursor(MOUSE *mouse);
void mouse_off_hand(MOUSE *mouse);

void Curinit();
void draw_mouse_arrow(int mx, int my);

void draw_cursor(int x, int y);
void hide_cursor(int x, int y);
void hide_cursor_grew(int x, int y);
void cursor(int x, int y);
void cursor_grew(int x, int y);
int mouse_location(int x1,int y1,int x2,int y2);
unsigned int Getpixel64k(int x, int y);
```

```
    #endif // MOUSE_H

/*****************************************************************/
    #ifndef __ARRANGE_H__
    #define __ARRANGE_H__

    typedef struct AcceptedOrder{
        int type; // 0:超市, 1:食品, 2:快递
        union {
            Order order;
            FoodOrder food;
            Deliver deliver;
        } data;
    } AcceptedOrder;//已接取的订单
    int arrange(int start_idx, struct AcceptedOrder acp_orders[], int
n_orders);
    void draw_arrange(int j, struct AcceptedOrder acp_orders[], int
    start_index, int best_i, int best_type);
    void calculate_centered_text(int rect_x1, int rect_y1, int rect_x2,
    int rect_y2, const char *text, int font_size, int *text_x, int
    *text_y);
    int Manhattan_Distance(int x1, int y1, int x2, int y2);

    #endif
/*****************************************************************/
    #ifndef _BUSI_DET_H
    #define _BUSI_DET_H

    void business_detail(int order_index,int index);
    void      draw_business_detail(OrderList      *OL      ,FoodOrder
    target_order[],int order_index,int page,int index);

    #endif
```

```
    #ifndef _BUSI_ORDER_H
    #define _BUSI_ORDER_H

    void business_order(int index);
    void draw_business_order(int page,OrderList *OL,FoodList *FL,int
index);

    #endif
/*****************************************************************/
    #ifndef __BUSI_H__
    #define __BUSI_H__

    void business(int user_pos);
    void draw_business();
    void press_type(int x);
    void draw_market();
    void draw_canteen();

    void choose_market(int mx,int my);
    int choose_canteen(int x, int y, int* last_index);

    #endif
/*****************************************************************/
    #ifndef __BUTTON_H__
    #define __BUTTON_H__

    typedef struct Button{
        int commmunity; // 社区编号
        int x1, y1, x2, y2; // 按钮坐标
        int index;          // 楼栋索引
        int number;        // 楼栋号
    } Button;
```

```
    void draw_button(int x);
    int press_button(int mx, int my, int cur_index, int cur_community);
    // cur_index: 当前按钮索引
    #endif
/*****************************************************************/
    #ifndef DE_ORDER_H
    #define DE_ORDER_H

    void de_order();
    void draw_de_order();

    #endif
/*****************************************************************/
    #ifndef FOOD_ORDER_H
    #define FOOD_ORDER_H

    #define F_LIST_INIT_SIZE 10 //线性表存储空间的初始分配量
    #define F_LISTINCEREMENT 1  //线性表存储空间的分配增量

    typedef struct FoodOrder{
        int id;//订单号
        int community;//用户地址
        int building;//楼栋号
        char user_name[12];//用户名
        char user_phone[12];//用户手机号
        char order_time[20];//下单时间
        FoodCart item[20];//购物车内物品信息
        int itemCount;//购物车内物品种类数量
        float total_amount;//总价
        int station;//食堂地址
        int destination;//送达地址
        int pick_up_location;//取餐地点
    } FoodOrder;//订单信息
```

```
    typedef struct FoodList
    {
        struct FoodOrder* elem;    //存储空间基址
        short length;  //当前长度，改为 short
        short listsize;  //当前存储空间容量，改为 short
    }FoodList;//订单线性表

    void food_order(int index);
    void draw_food_order(int page,float *sum);

    void ReadAllFood(FoodList *OL);
    void DestroyFList(FoodList*OL);
    int FoodOrder_pos(FoodList OL,FoodOrder Foodorders);
    void FListInsert(FoodList*OL,struct FoodOrder e);
    int save_food(FoodOrder FoodOrders);

    #endif
/*****************************************************************/
    #ifndef __MY_ACP_H
    #define __MY_ACP_H

    void rider_accept(OrderList *OL, FoodList *FL, DeliverList *DL,
        int type, int local_index, int page );

    void my_accept_order();

    void draw_my_accept();

    #endif
/*****************************************************************/
    #ifndef __RIDE_H__
    #define __RIDE_H__
```

```
    void rider(int user_pos);
    void draw_rider();
    void press3(int x);

    #endif
/*****************************************************************/
    #ifndef ROUTE_H
    #define ROUTE_H
    #include "arrange.h"
    typedef struct Node {
        int x, y;
        int adj_nodes[6];
        int distance[6];
        int num_of_adj_nodes;
        char name [20];//节点名称
    } Node;

    // 在 route 函数外部定义任务状态结构体
    typedef struct  RouteState {
        int picked[4];       // 记录 4 个订单的取餐状态
        int delivered[4];    // 记录 4 个订单的送餐状态
        int remaining;       // 剩余任务数
        int current_pos;     // 当前位置索引
        int next_pos;
        int next_type;       // 0 = 取餐, 1 = 送餐
    } RouteState;

    int random_int(int min, int max);
    void route( AcceptedOrder acp_orders[], int n_orders);

    int dijkstra(Node *start, Node *end, int j);
```

```
    void draw_route();
    #endif
/*****************************************************************/
    #ifndef __DRAW_SHAPE_H__
    #define __DRAW_SHAPE_H__

    void Draw_Rounded_Rectangle(int x1, int y1, int x2, int y2,
                    int radius, int width,unsigned int color);

    void DrawArcWide(int xc, int yc, int radius, int width,
            unsigned   int   color,   float   start_angle_deg,   float
end_angle_deg);

    void Fill_Rounded_Rectangle(int x1, int y1, int x2, int y2,
                               int radius, unsigned int color);



    #endif
/*****************************************************************/
    #ifndef __USER_CART_H__
    #define __USER_CART_H__

    void user_cart();
    void   draw_user_cart(CartItem   carts[],   int   cartCount,   int
page,float *sum);
    void draw_user_cart_quantity(CartItem carts[], int index, int y);
     void AddSub_cart(int mx, int my, CartItem carts[], int* itemCount,
     int currentPage,float *sum);
    #endif
/*****************************************************************/
    #ifndef __USER_DE_H__
    #define __USER_DE_H__
```

```
#define D_LIST_INIT_SIZE 10 //线性表存储空间的初始分配量
#define D_LISTINCEREMENT 1  //线性表存储空间的分配增量

typedef struct Deliver
{
    int id; // 订单号
    int type; // 快递类型
    char name[10]; // 用户名
    char number[12];//手机号
    int community; //地址 1 东区 2 西区 3 南区 4 紫菘 5 韵苑
    int building;//楼栋号
    int destination;//送达地址
    char time[20]; // 下单时间
    char code[10]; //取件码
    int company; // 快递公司
    int station; // 站点
    int index;
    int total_cnt;
    int acp_count;
} Deliver;

typedef struct DeliverList
{
    struct Deliver* elem;    //存储空间基址
    short length;  //当前长度，改为 short
    short listsize;  //当前存储空间容量，改为 short
}DeliverList;//订单线性表

typedef struct Company{//快递公司
    char name[20];//名称
}Company;

typedef struct Station{//站点
```

```
    char name[20];//名称
}Station;

void user_deliver();
void draw_user_deliver();
int choose_company(int x, int y, int* last_index);
int choose_station(int x, int y, int* last_index_station);

void deliver_input(char *deliver_code,int bar_x1,int bar_y1,int
bar_x2,int bar_y2);

int Deliver_pos(DeliverList DL,Deliver delivers);
void DListInsert(DeliverList*DL,Deliver delivers);
int save_Deliver(Deliver delivers);
void DestroyDList(DeliverList*DL);
void ReadAllDeliver(DeliverList *DL);

#endif
/*****************************************************************/
#ifndef USER_FOOD_H
#define USER_FOOD_H

typedef struct Food {
    int id; // 食物 ID
    int station; // 食堂编号
    char name[50]; // 食物名称
    float price; // 食物价格
    char photo[50]; // 食物图片路径
    int quantity; // 食物数量
} Food;

typedef struct FoodCart {
    int id;//商品 id
```

```
        int type;//种类
        char name[20];//商品名称
        float price;//价格
        char photo[50];
        int quantity;//数量
        int index_in_foods;//在商品数组中的索引，即 cnt
    } FoodCart;//购物车内的商品

    typedef struct ShoppingFood{
        FoodCart* items;//购物车内商品
        int itemCount;//购物车内商品种类数量
        int capacity;//购物车容量
    } ShoppingFood;//购物车整体

    void user_food(int index);
    void draw_user_food(int index);
    void draw_food_quantity(Food foods[]);
    void AddSub_food(int mx, int my, int foodCount, Food foods[],
    FoodCart food_carts[], int* itemCount);
    void addToCart_food(Food f, FoodCart food_carts[], int
*itemCount,int index);
    void removeFromCart_food(Food f, FoodCart food_carts[], int
*itemCount);
    #endif
/*****************************************************************/
    #ifndef __USER_ORDER_H__
    #define __USER_ORDER_H__

    #define O_LIST_INIT_SIZE 10 //线性表存储空间的初始分配量
    #define O_LISTINCEREMENT 1  //线性表存储空间的分配增量

    typedef struct Order{
        int id;//订单号
```

```
    int community;//用户地址
    int building;//楼栋号
    int pick_up_location;//取餐地点
    int destination;
    char user_name[12];//用户名
    char user_phone[12];//用户手机号
    char order_time[20];//下单时间
    CartItem item[20];//购物车内物品信息
    int itemCount;//购物车内物品种类数量
    float total_amount;//总价
} Order;//订单信息

typedef struct OrderList
{
   struct Order* elem;    //存储空间基址
   short length;  //当前长度，改为 short
   short listsize;  //当前存储空间容量，改为 short
}OrderList;//订单线性表

void user_order();
void draw_user_order(int page);
void draw_info();

char* get_current_time();

void InitOList(OrderList*OL);
void ReadAllOrder(OrderList *OL);
void DestroyOList(OrderList*OL);
int Order_pos(OrderList OL,Order orders);
void OListInsert(OrderList*OL,struct Order e);
int save_order(Order orders);

#endif
```

```
/*****************************************************************/
    #ifndef __USER_RE_H__
    #define __USER_RE_H__

    void user_register();
    void draw_register();
    void press(int x);

    #endif
/*****************************************************************/

    #ifndef __USER_SHOP_H__
    #define __USER_SHOP_H__

    typedef struct CartItem{
        int id;//商品 id
        int type;//种类
        char name[20];//商品名称
        float price;//价格
        char photo[50];
        int quantity;//数量
        int index_in_products;//在商品数组中的索引，即 cnt
    } CartItem;//购物车内的商品

    typedef struct ShoppingCart{
        CartItem* items;//购物车内商品
        int itemCount;//购物车内商品种类数量
    } ShoppingCart;//购物车整体

    typedef struct Product
    {
        int id;
        int type;//种类
```

```
    char name[20];
    float price;
    char photo[50];
    int quantity;//数量

} Product;//商品


void user_shop();
void draw_user_shop(Product products[],int productCount,int currentpage);
void draw_user_shop_quantity(Product products[],int productCount,int currentpage);
void init_Products(int *productCount);
void AddSub(int mx, int my, int productCount, Product products[], CartItem carts[], int *cartCount, int currentPage);
void addToCart(Product p, CartItem carts[], int *cartCount,int index);
void removeFromCart(Product p, CartItem carts[], int *cartCount);

void draw_sort();

#endif
/*****************************************************************/
#ifndef __USER_TAKE_H__
#define __USER_TAKE_H__

typedef struct Canteen {
    int id; // 食堂ID
    char name[50]; // 食堂名称
}Canteen;

void user_takeout();
```

```
    void draw_user_takeout();
    int press_canteen(int mx, int my);

    #endif
/*****************************************************************/
    #ifndef __USER_H__
    #define __USER_H__

    void user(int user_pos);
    void draw_user();
    void press_func(int x);
    void   number_input(char   *number,int   bar_x1,int   bar_y1,int
bar_x2,int bar_y2);

    #endif
/*****************************************************************/
    #ifndef __DRAW_H__
    #define __DRAW_H__

    int welcome();
    void draw_basic();
    void draw_about_us();
    void draw_about_product();

    #endif
/*****************************************************************/
```

4. 源文件

```
    #include "all_func.h"
    void business_detail(int order_index,int index) {

        OrderList OL = {0};
        FoodList FL = {0};
```

```
        FoodOrder target_order[12];

        int page = 0;// 初始页码
        int totalPage; // 总页数
        int i;
        int cnt=0;

        ReadAllOrder(&OL); // 读取订单列表
        ReadAllFood(&FL); // 读取食物列表



        //进行区分订单来源操作并且计算总页数
        if(index==0){
            totalPage =(OL.elem[order_index].itemCount  - 6 + 11 ) /
12 + 1 ; // 总页数(向上取整)
        }else {
            for(i=0;i<FL.length;i++)
            {
               if(FL.elem[i].station==index)//找到对应的订单
               {
                  target_order[cnt]=FL.elem[i];//创建 target_order 数组,
此时这个 target 和 OL.elem 是并列的
                   cnt++;


               }
            }
            totalPage =(target_order[order_index].itemCount - 6 + 11 )
/ 12 + 1 ; // 总页数(向上取整)
        }


        mouse_off_arrow(&mouse);
```

```
        draw_business_detail(&OL,target_order,
order_index ,page,index); // 绘制订单详情页面

        mouse_on_arrow(mouse);

        while(1){
           mouse_show_arrow(&mouse);

           if(mouse_press(40, 113, 160, 163)==1)
           {
               DestroyOList(&OL); // 释放订单列表内存
               DestroyFList(&FL); // 释放食物列表内存
               return;
              //business_order();//返回
           }
           else if (mouse_press(220, 700, 340, 750) == 1)
           {
               if (page > 0) {
                   page--;
                   draw_business_detail(&OL,target_order,
order_index ,page,index);
               } else {
                   // 提示：已是第一页
                   PrintCC(550, 25, "已是第一页", HEI, 24, 1, lightred);
                  delay(500);
                  bar1(550, 25, 700, 60, white);
               }
           }
           else if (mouse_press(420, 700, 540, 750) == 1)
           {
              if (page < totalPage - 1) {
                  page++;
                  draw_business_detail(&OL,target_order,
```

```
order_index ,page,index);
              } else {
                  // 提示：已是最后一页
                  PrintCC(550,  25,  "已是最后一页",  HEI,  24,  1,
lightred);
                  delay(500);
                  bar1(550, 25, 700, 60, white);
              }
           }
        }
    }

    void      draw_business_detail(OrderList      *OL      ,FoodOrder
target_order[],int order_index,int page,int index) {
        int i;
        Order currentOrder ;
        FoodOrder currentFood;

        char current_time[20]; // 获取当前时间
        char time_str[100]; // 打印下单时间
        char user_name[100]; // 打印用户名
        char user_phone[100]; // 打印用户手机号
        char order_number; // 打印订单号
        char address[100]; // 打印用户地址
        int startIdx = 0;// 起始商品索引
        int itemsPerPage = 0;// 每页商品数量
        int endIdx = 0;// 结束商品索引
        int item_y = 0;// 商品框的y坐标

        float total_amount = 0.0; // 总金额
        char total_str[20]; // 总金额字符串
        int fullPageItemCount = 0; // 满页商品数量
```

```
        bar1(200, 0, 1024, 768, white); // 清空屏幕

        // 分页按钮
        Draw_Rounded_Rectangle(220, 700, 340, 750, 25, 1, deepblue);
// 上一页
        Draw_Rounded_Rectangle(420, 700, 540, 750, 25, 1, deepblue);
// 下一页
        PrintCC(245, 715, "上一页", HEI, 24, 1, deepblue);
        PrintCC(445, 715, "下一页", HEI, 24, 1, deepblue);

        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5, 1, deepblue);
// 开始备货
        PrintCC(850, 715, "开始备货", HEI, 24, 1, deepblue);

        if(index==0)
        {
            currentOrder = OL->elem[order_index]; // 当前订单
            strcpy(current_time, currentOrder.order_time);

            sprintf(time_str, "下单时间：%s", current_time);
            sprintf(user_name, "用户名：%s", currentOrder.user_name);
            sprintf(user_phone, "手机号：%s", currentOrder.user_phone);
        }
        else
        {
            currentFood = target_order[order_index]; // 当前订单
            strcpy(current_time, currentFood.order_time);

            sprintf(time_str, "下单时间：%s", current_time);
            sprintf(user_name, "用户名：%s", currentFood.user_name);
            sprintf(user_phone, "手机号：%s", currentFood.user_phone);
        }
```

```
        // 页头信息只在第一页显示
        if (page == 0) {
            char order_number_str[20]; // 订单号字符串
            char community[50]; // 社区字符串
            char building[50]; // 楼栋字符串


            if(index==0)//超市订单
            {
                sprintf(order_number_str, "订单号:%d", currentOrder.id);
// 订单号
                sprintf(address,       "     地     址     :      %s",
node[currentOrder.destination].name); // 用户地址
            }
            else //食堂订单
            {
                sprintf(order_number_str, "订单号:%d", currentFood.id);
// 订单号
                sprintf(address,       "     地     址     :      %s",
node[currentFood.destination].name); // 用户地址
                PrintCC(750,250,  canteen[currentFood.station-1].name,
HEI, 24, 1, black);//显示食堂名称
            }

            PrintText(250, 50, order_number_str, HEI, 24, 1, black);
            PrintText(250, 100, time_str, HEI, 24, 1, black);
            PrintText(250, 150, user_name, HEI, 24, 1, black);
            PrintText(250, 200, user_phone, HEI, 24, 1, black);

            PrintText(250, 250, address, HEI, 24, 1, black);

            // 表头
```

```
            PrintCC(250, 300, "商品详情：", HEI, 24, 1, black);
            PrintCC(750, 300, "数量：", HEI, 24, 1, black);
            PrintCC(900, 300, "金额：", HEI, 24, 1, black);
            PrintText(250, 320, "------------------------------", HEI,
32, 1, black);// 分隔线

            startIdx = 0;
            itemsPerPage = 6;
        } else {// 其他页
            startIdx = 6 + (page - 1) * 12;
            itemsPerPage = 12;
        }

        endIdx = startIdx + itemsPerPage;

        if (index==0)//超市订单
        {
            if (endIdx > currentOrder.itemCount)// 防止越界
                endIdx = currentOrder.itemCount;
        }
        else //食堂订单
        {
            if (endIdx > currentFood.itemCount)// 防止越界
                endIdx = currentFood.itemCount;
        }

        item_y = (page == 0) ? 350 : 50;
        for (i = startIdx; i < endIdx; i++) {
            char total_str[50]; // 商品总价
            char quantity_str[20]; // 商品数量

            if(index==0)//超市订单
            {
```

```
                int quantity = currentOrder.item[i].quantity; // 商品数
量
                float price = currentOrder.item[i].price; // 商品价格

                sprintf(total_str, "%.2f", price * quantity);
                sprintf(quantity_str, "x%d", quantity);

                PrintCC(250, item_y, currentOrder.item[i].name, HEI, 24,
1, black); // 商品名
            }
            else //食堂订单
            {
                int quantity = currentFood.item[i].quantity; // 商品数
量
                float price = currentFood.item[i].price; // 商品价格

                sprintf(total_str, "%.2f", price * quantity);
                sprintf(quantity_str, "x%d", quantity);

                PrintCC(250, item_y, currentFood.item[i].name, HEI, 24,
1, black); // 商品名
            }

            PrintText(750, item_y, (unsigned char*)quantity_str, HEI,
24, 1, black);// 商品数量
            PrintText(900, item_y, (unsigned char*)total_str, HEI, 24,
1, black);// 商品总价

            item_y += 50;
        }

        // 判断是否需要在此页显示总金额（当前页没有满）
        fullPageItemCount = (page == 0) ? 6 : 12;// 第一页显示 6 个商品，
```

其余页显示 12 个商品

```
        if (index==0)
        {
            if ((endIdx - startIdx) < fullPageItemCount||endIdx==currentOrder.itemCount) {// 当前页商品数量不满一页或最后一个商品刚好满页都要打印出总金额
                //如果不是最后一个商品但是满页就不打印总金额
                // 打印分隔线
                PrintText(250, item_y - 30, "------------------------------", HEI, 32, 1, black);

                // 计算总金额
                total_amount = 0.0;
                for (i = 0; i < currentOrder.itemCount; i++) {
                    int quantity = currentOrder.item[i].quantity; // 商品数量
                    float price = currentOrder.item[i].price; // 商品价格
                    total_amount += price * quantity;
                }

                sprintf(total_str, "总金额：%.2f 元", total_amount);
                PrintText(750, item_y + 10, total_str, HEI, 24, 1, black);

            }
        }
        else
        {
            if ((endIdx - startIdx) < fullPageItemCount||endIdx==currentFood.itemCount) {// 当前页商品数量不满一页或最后一个商品刚好满页都要打印出总金额
```

```
                //如果不是最后一个商品但是满页就不打印总金额
                // 打印分隔线
                PrintText(250, item_y - 30, "------------------------
------", HEI, 32, 1, black);

                // 计算总金额
                total_amount = 0.0;
                for (i = 0; i < currentFood.itemCount; i++) {
                    int quantity = currentFood.item[i].quantity; // 商
品数量
                    float price = currentFood.item[i].price; // 商品价
格
                    total_amount += price * quantity;
                }

                sprintf(total_str, "总金额：%.2f 元", total_amount);
                PrintText(750, item_y + 10, total_str, HEI, 24, 1,
black);

            }
        }

    }
/*****************************************************************/
    #include "all_func.h"

    void business_order(int index){
        int page = 0; // 当前页码

        OrderList OL = {0};
        FoodList FL = {0};

        ReadAllOrder(&OL); // 读取订单列表
```

```
ReadAllFood(&FL); // 读取食品列表

mouse_off_arrow(&mouse);

draw_business_order(page,&OL,&FL,index);

mouse_on_arrow(mouse);

while(1){
    mouse_show_arrow(&mouse);

    if(mouse_press(40, 113, 160, 163)==1)
    {
        DestroyOList(&OL); // 释放订单列表内存
        DestroyFList(&FL); // 释放食品列表内存
        return;
        //business(users.pos);
    }
    else if (mouse_press(220, 700, 340, 750) == 1)
    {
        if (page > 0) {
            page--;
            draw_business_order(page,&OL,&FL,index);
        } else {
            // 提示：已是第一页
            PrintCC(550, 35, "已是第一页", HEI, 24, 1, lightred);
            delay(500);
            bar1(550, 35, 700, 60, white);
        }
    }
    else if (mouse_press(420, 700, 540, 750) == 1)
    {
        if ((page + 1) * 5 < OL.length) {
```

```
                    page++;
                    draw_business_order(page,&OL,&FL,index);
                } else {
                    // 提示：已是最后一页
                        PrintCC(550, 35, "已是最后一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(550, 35, 700, 60, white);
                }
            }
            else if(mouse_press(200, 0, 1024, 680) == 1) {

                int order_index = (mouse.y - 25) / 120 + page * 5; //
根据点击位置计算订单索引
                //这个 order_index 是基于展示的页面计算的索引，如果是 food,
就是 target 里的索引，因为是从 target 中展示的
                //order_index 对二者都适用
                    //传入 index 是为了区分超市和不同食堂，如果是食堂就要用
target 来展示
                MouseGet(&mouse);
                business_detail(order_index,index); // 显示订单详情

                //return 后从这开始
                mouse_off_arrow(&mouse);
                bar1(200, 0, 1024, 768, white); // 清除订单详情界面残留
                 draw_business_order(page,&OL,&FL,index); // 重新绘制订
单列表
                mouse_on_arrow(mouse);
            }
        }
    }

    void draw_business_order(int page,OrderList *OL,FoodList *FL,int
```

```
index){

        int i;
        int cnt=0;
        int y_offset = 25; // 初始 Y 轴偏移

        FoodOrder target_order[12];

        // 每页最多显示 5 个订单
        int start_index = page * 5; // 当前页的起始订单索引
        int end_index = start_index + 5; // 当前页的结束订单索引

        if(index==0){//
            if (end_index > OL->length)
            {
                end_index = OL->length; // 防止越界
            }
        }
        else
        {
            for(i=0;i<FL->length;i++)
            {
                if(FL->elem[i].station==index)//找到对应的订单
                {
                    target_order[cnt]=FL->elem[i];
                    cnt++;

                }
            }
            if (end_index > cnt)
            {
                end_index = cnt; // 防止越界
            }
```

```
        }

        bar1(200, 0, 1024, 768,white);

        Fill_Rounded_Rectangle(40, 113, 160, 163, 25,white);//返回填色
         Draw_Rounded_Rectangle(40, 113, 160, 163, 25, 1,deepblue);//
返回//圆角按钮，字的 x+35，y+15
        PrintCC(75,128,"返回",HEI,24,1,deepblue);


        for (i = start_index; i < end_index; i++) {
            char order_id[10]; // 订单 ID 字符串
            char user_info[16]; // 用户信息字符串
            char total_price[10];// 总价字符串

            if(index==0){
                Order order = OL->elem[i]; // 获取当前订单

                // 绘制订单框
                 Draw_Rounded_Rectangle(220, y_offset, 1000, y_offset
+ 100, 30, 1, 0x6B4D);

                // 显示订单简略信息
                sprintf(order_id, "订单%d", i + 1);
                 PrintText(235, y_offset + 15, order_id, HEI, 24, 1,
0x0000);

                sprintf(user_info, "下单人：%s", order.user_name);
                 PrintText(235, y_offset + 65, user_info, HEI, 24, 1,
0x0000);
```

```
                sprintf(total_price, "总价：%.2f", order.total_amount);
                PrintText(800, y_offset + 15, total_price, HEI, 24, 1,
0x0000);

                 PrintText(665, y_offset + 65, order.order_time, HEI,
24, 1, 0x0000);

                y_offset += 120; // 每个订单框之间的间距
            }else if(index>0)
            {
                    FoodOrder food_order = target_order[i];// 获取当前
订单

                   // 绘制订单框
                         Draw_Rounded_Rectangle(220, y_offset, 1000,
y_offset + 100, 30, 1, 0x6B4D);

                   // 显示订单简略信息
                   sprintf(order_id, "订单%d", i + 1);
                    PrintText(235, y_offset + 15, order_id, HEI, 24,
1, 0x0000);

                                sprintf(user_info, "下单人：%s",
food_order.user_name);
                   PrintText(235, y_offset + 65, user_info, HEI, 24,
1, 0x0000);


                               sprintf(total_price, "总价：%.2f",
food_order.total_amount);
                    PrintText(800, y_offset + 15, total_price, HEI,
24, 1, 0x0000);
```

```
                                PrintText(665,  y_offset  +  65,
food_order.order_time, HEI, 24, 1, 0x0000);

                    y_offset += 120; // 每个订单框之间的间距
                }

            }

        // 绘制翻页按钮
        Draw_Rounded_Rectangle(220, 700, 340, 750, 25, 1, deepblue);
// 上一页
        Draw_Rounded_Rectangle(420, 700, 540, 750, 25, 1, deepblue);
// 下一页
        PrintCC(245, 715, "上一页", HEI, 24, 1, deepblue);
        PrintCC(445, 715, "下一页", HEI, 24, 1, deepblue);
    }
/*****************************************************************/
    #include "all_func.h"

    void business(int user_pos){

        UserList UL = {0};
        USER currentUser;
        int shop_type=0;//商店类型，1 为超市，2 为餐厅
        int code[12]={0};//绑定码
        int state=0;//状态，0 为未绑定，1 为已输入绑定码，2 为已绑定
        int page=0;//0 为未选择，1 为超市，2 为餐厅
        int last_canteen_index = -1; // 记录上一个被按下的按钮
        int temp_index = -1; // 临时变量,选择食堂/超市时不保存
        int index=-1;//食堂/超市编号


        ReadAllUser(&UL); // 读取用户列表
```

```
        currentUser=UL.elem[user_pos];// 获取当前用户信息

        DestroyUList(&UL); // 释放用户列表空间

        mouse_off_arrow(&mouse);

        draw_business();

        mouse_on_arrow(mouse);

        while(1){
            mouse_show_arrow(&mouse);

            if(mouse_press(40, 113, 160, 163)==1)
            {
                DestroyUList(&UL); // 释放用户列表空间
                return;
                //welcome();//首页
            }
            else if(mouse_press(430, 110, 650, 160)==1)//输入手机号
            {
                 number_input(currentUser.number, 435, 115, 645, 155);
// 输入手机号
            }
            else if(mouse_press(710, 110, 830, 160)==1)//保存手机号
            {
                if(strlen(currentUser.number)==11)
                {
                    save_user(currentUser);
                    PrintCC(700,50,"保存成功",HEI,24,1,lightred);
                    delay(500);
                    bar1(600,50,1024,100,white);
```

```
        }
        else
        {
            PrintCC(700,50,"长度不合法",HEI,24,1,lightred);
            delay(500);
            bar1(600,50,1024,100,white);
        }
    }



    //输入绑定码后但未选择店铺
    if(state==1)
    {
        if(mouse_press(490, 260, 610, 310)==1)
        {
            press_type(1);//选择为超市经营者
            shop_type=1;//超市
            page=1;


        }
        else if(mouse_press(670, 260, 790, 310)==1)
        {
            press_type(2);//选择为餐厅经营者
            shop_type=2;//餐厅
            page=2;
        }
        else if(mouse_press(205, 325, 1024, 680)==1)//选择具体
食堂/超市
        {
            MouseGet(&mouse);
            mouse_off_arrow(&mouse);
            if(page==1)//超市
```

```
                {
                    temp_index=0;//选择超市
                    choose_market(mouse.x, mouse.y);
                }

                if(page==2)//餐厅
                {
                       temp_index=choose_canteen(mouse.x, mouse.y,
&last_canteen_index);
                }
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(800, 700, 1000, 750)==1)//确认绑定
            {
                index=temp_index;//保存
                bar1(600,50,1024,100,white);
                PrintCC(700,50,"绑定成功",HEI,24,1,lightred);
                delay(500);
                bar1(600,50,1024,100,white);
                state=2;//已绑定
                currentUser.state=index;//已绑定
                save_user(currentUser);

            }else if(mouse_press(40, 602, 160, 652)==1)//查看订单
            {
                bar1(600,50,1024,100,white);
                          PrintCC(750,50,"请先选择绑定的店铺
",HEI,24,1,lightred);
                delay(500);
                bar1(600,50,1024,100,white);
            }

        }
```

```
            //未绑定
            if(state==0&&currentUser.state==-1)
            {
                if(mouse_press(490, 260, 610, 310)==1||
                   mouse_press(670, 260, 790, 310)==1||
                   mouse_press(40, 602, 160, 652)==1)
                {
                                PrintCC(800,50," 请 先 进 行 绑 定 操 作
",HEI,24,1,lightred);
                    delay(500);
                    bar1(600,50,1024,100,white);

                }
                 else if(mouse_press(430, 185, 650, 235)==1)//输入绑定
码
                {
                     number_input(code, 435, 190, 645, 230); // 输入绑
定码
                }
                else if(mouse_press(710, 185, 830, 235)==1)//确认绑定
                {
                    if(strcmp(code,"111")==0)
                    {
                        PrintCC(800,50,"验证成功",HEI,24,1,lightred);
                        delay(500);
                        bar1(600,50,1024,100,white);
                        state=1;//已绑定
                    }
                    else
                    {
                        PrintCC(800,50,"验证失败",HEI,24,1,lightred);
                        delay(500);
```

```
                    bar1(600,50,1024,100,white);
                }
            }
        }

        //已绑定
        if(state==2||currentUser.state!=-1)
        {
            if(mouse_press(40, 602, 160, 652)==1)//查看订单
            {
                if(strlen(currentUser.number) == 0)//如果没有输入手
机号
                {
                                PrintCC(800,50," 请 先 输 入 手 机 号
",HEI,24,1,lightred);
                    delay(500);
                    bar1(600,50,1024,100,white);
                }
                else
                {
                                        if(currentUser.state!=-1)
index=currentUser.state;//如果已经存在文件里直接读取
                    DestroyUList(&UL); // 释放用户列表空间
                    business_order(index);//商家订单页面

                    //return 后从这开始
                    mouse_off_arrow(&mouse);
                    bar1(200, 0, 1024, 768, white); // 清除注册界面
残留
                    draw_business();
                    mouse_on_arrow(mouse);
                }
```

```
                }
                if(mouse_press(205, 185, 1024,  680)==1)
                {
                    bar1(600,50,1024,100,white);
                    PrintCC(650,50,"已绑定，无法更改",HEI,24,1,lightred);
                }
                if(mouse_press(430, 110, 650, 160)==1)//输入手机号
                {
                      number_input(currentUser.number, 435, 115, 645,
155); // 输入手机号
                }
                 else if(mouse_press(710, 110, 830, 160)==1)//保存手机
号
                {
                    if(strlen(currentUser.number)==11)
                    {
                        save_user(currentUser);
                        PrintCC(700,50,"保存成功",HEI,24,1,lightred);
                        delay(500);
                        bar1(600,50,1024,100,white);
                    }
                    else
                    {
                         PrintCC(700,50,"长度不合法",HEI,24,1,lightred);
                        delay(500);
                        bar1(600,50,1024,100,white);
                    }
                }
            }
        }
    }

    void draw_business()
```

```
    {
        bar1(0, 0, 1024, 768,white);
        bar1(0, 0, 200, 768,deepblue);

        Fill_Rounded_Rectangle(40, 113, 160, 163, 25,white);//返回填色
        Draw_Rounded_Rectangle(40, 113, 160, 163, 25, 1,deepblue);//返回//圆角按钮，字的 x+35，y+15
        Fill_Rounded_Rectangle(40, 602, 160, 652, 25,white);//确认填色
        Draw_Rounded_Rectangle(40, 602, 160, 652, 25, 1,deepblue);//确认

        Draw_Rounded_Rectangle(490, 260, 610, 310, 25, 1,deepblue);//超市按钮
        Draw_Rounded_Rectangle(670, 260, 790, 310, 25, 1,deepblue);//餐厅按钮

        Draw_Rounded_Rectangle(430, 110, 650, 160, 5, 1,deepblue);//手机号输入框
        Draw_Rounded_Rectangle(710, 110, 830, 160, 25, 1,deepblue);//保存手机号按钮
        Draw_Rounded_Rectangle(430, 185, 650, 235, 5, 1,deepblue);//绑定码输入框
        Draw_Rounded_Rectangle(710, 185, 830, 235, 25, 1,deepblue);//确认按钮
        //Draw_Rounded_Rectangle(430, 330, 650, 375, 5, 1,deepblue);//查看订单按钮

        PrintCC(75,128,"返回",HEI,24,1,deepblue);
        PrintCC(52,617,"查看订单",HEI,24,1,deepblue);

        PrintCC(250,50,"当前账号类型为：商家",HEI,24,1,deepblue);

        PrintCC(250,125,"请输入手机号：",HEI,24,1,deepblue);
```

```
        PrintCC(745,125,"保存",HEI,24,1,deepblue);
        PrintCC(250,200,"请输入绑定码：",HEI,24,1,deepblue);
        PrintCC(745,200,"确认",HEI,24,1,deepblue);

        PrintCC(250,275,"请选择店铺种类：",HEI,24,1,deepblue);
        PrintCC(525,275,"超市",HEI,24,1,deepblue);
        PrintCC(705,275,"餐厅",HEI,24,1,deepblue);

    }

    void press_type(int x){
        mouse_off_arrow(&mouse);
        switch (x)
        {
            case 1:{//进入选择超市页面
                    Fill_Rounded_Rectangle(490,  260,  610,  310,  25,
deepblue);//超市
                    Draw_Rounded_Rectangle(490,  260,  610,  310,  25,
1,deepblue);//超市
                PrintCC(525,275,"超市",HEI,24,1,white);
                    Fill_Rounded_Rectangle(670,  260,  790,  310,  25,
white);//餐厅
                    Draw_Rounded_Rectangle(670,  260,  790,  310,  25,
1,deepblue);//餐厅
                PrintCC(705,275,"餐厅",HEI,24,1,deepblue);

                draw_market();

                break;
            }
            case 2:{//进入选择餐厅页面
                    Fill_Rounded_Rectangle(490,  260,  610,  310,  25,
white);//超市
```

```
                    Draw_Rounded_Rectangle(490,  260,  610,  310,  25,
1,deepblue);//超市
                PrintCC(525,275,"超市",HEI,24,1,deepblue);
                    Fill_Rounded_Rectangle(670,  260,  790,  310,  25,
deepblue);//餐厅
                    Draw_Rounded_Rectangle(670,  260,  790,  310,  25,
1,deepblue);//餐厅
                PrintCC(705,275,"餐厅",HEI,24,1,white);

                draw_canteen();

                break;
            }

        }
        mouse_on_arrow(mouse);
    }

    void draw_market(){
        bar1(205, 325, 1024, 768, white);

        Draw_Rounded_Rectangle(250, 330, 250+185, 330+50, 5,1,0x0235);
        PrintCC(250+17,330+13,"韵苑喻园超市",HEI,24,1,0x0235);

        Draw_Rounded_Rectangle(500, 330, 500+185, 330+50, 5,1,0x0235);
        PrintCC(500+17,330+13,"沁苑喻园超市",HEI,24,1,0x0235);

        Draw_Rounded_Rectangle(750, 330, 750+185, 330+50, 5,1,0x0235);
        PrintCC(750+17,330+13,"紫菘喻园超市",HEI,24,1,0x0235);

        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5, 1, deepblue);
// 确认绑定
        PrintCC(870, 715, "确定", HEI, 24, 1, deepblue);
```

```
        PrintCC(600,50,"仅能绑定一次，请慎重操作",HEI,24,1,lightred);

    }

    void draw_canteen(){
        int i,j;
        int cnt=0;
        bar1(205, 325, 1024, 768, white);

        //打印食堂名称
        for(i=0;i<6;i++){
            for(j=0;j<3;j++){
                if(cnt==17) break;
                        Draw_Rounded_Rectangle(250+250*j,  330+60*i,
250+250*j+185, 330+60*i+50, 5,1,deepblue);
                 PrintCC(250+250*j+17,330+60*i+13,canteen[cnt].name,HE
I,24,1,deepblue);
                cnt++;
            }
        }

        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5, 1, deepblue);
// 确认绑定
        PrintCC(870, 715, "确定", HEI, 24, 1, deepblue);

        PrintCC(600,50,"仅能绑定一次，请慎重操作",HEI,24,1,lightred);

    }

    void choose_market(int mx,int my){
        if(mx>=250&&mx<=250+185&&my>=330&&my<=330+50)//韵苑
        {
```

```
                Fill_Rounded_Rectangle(250,  330,  250+185,  330+50,
5,deepblue);
                Draw_Rounded_Rectangle(250,  330,  250+185,  330+50,
5,1,deepblue);
           PrintCC(250+17,330+13,"韵苑喻园超市",HEI,24,1,white);

           Fill_Rounded_Rectangle(500, 330, 500+185, 330+50, 5,white);
                Draw_Rounded_Rectangle(500,  330,  500+185,  330+50,
5,1,deepblue);
           PrintCC(500+17,330+13,"沁苑喻园超市",HEI,24,1,deepblue);

           Fill_Rounded_Rectangle(750, 330, 750+185, 330+50, 5,white);
                Draw_Rounded_Rectangle(750,  330,  750+185,  330+50,
5,1,deepblue);
           PrintCC(750+17,330+13,"紫菘喻园超市",HEI,24,1,deepblue);

       }
       else if(mx>=500&&mx<=500+185&&my>=330&&my<=330+50)//沁苑
       {
           Fill_Rounded_Rectangle(250, 330, 250+185, 330+50, 5,white);
                Draw_Rounded_Rectangle(250,  330,  250+185,  330+50,
5,1,deepblue);
           PrintCC(250+17,330+13,"韵苑喻园超市",HEI,24,1,deepblue);

                Fill_Rounded_Rectangle(500,  330,  500+185,  330+50,
5,deepblue);
                Draw_Rounded_Rectangle(500,  330,  500+185,  330+50,
5,1,deepblue);
           PrintCC(500+17,330+13,"沁苑喻园超市",HEI,24,1,white);

           Fill_Rounded_Rectangle(750, 330, 750+185, 330+50, 5,white);
                Draw_Rounded_Rectangle(750,  330,  750+185,  330+50,
5,1,deepblue);
```

```
            PrintCC(750+17,330+13,"紫菘喻园超市",HEI,24,1,deepblue);
        }
        else if(mx>=750&&mx<=750+185&&my>=330&&my<=330+50)//紫菘
        {
            Fill_Rounded_Rectangle(250, 330, 250+185, 330+50, 5,white);
                 Draw_Rounded_Rectangle(250,  330,  250+185,  330+50,
5,1,deepblue);
            PrintCC(250+17,330+13,"韵苑喻园超市",HEI,24,1,deepblue);

            Fill_Rounded_Rectangle(500, 330, 500+185, 330+50, 5,white);
                 Draw_Rounded_Rectangle(500,  330,  500+185,  330+50,
5,1,deepblue);
            PrintCC(500+17,330+13,"沁苑喻园超市",HEI,24,1,deepblue);

                 Fill_Rounded_Rectangle(750,  330,  750+185,  330+50,
5,deepblue);
                 Draw_Rounded_Rectangle(750,  330,  750+185,  330+50,
5,1,deepblue);
            PrintCC(750+17,330+13,"紫菘喻园超市",HEI,24,1,white);
        }
    }

    int choose_canteen(int x, int y, int* last_index) {
        int i, j;
        int index = -1;

        for (i = 0; i < 6; i++) {
            for (j = 0; j < 3; j++) {
                int x1 = 250 + 250 * j;
                int y1 = 330 + 60 * i;
                int x2 = x1 + 185;
                int y2 = y1 + 50;
```

```
            if(i * 3 + j >= 17) break; // 超出食堂数量则退出

            if (x >= x1 && x <= x2 && y >= y1 && y <= y2) {
                index = i * 3 + j;

                // 恢复上一个按钮
                if (*last_index != -1 && *last_index != index) {
                    int pre_row = *last_index / 3;
                    int pre_col = *last_index % 3;
                    int pre_x1 = 250 + 250 * pre_col;
                    int pre_y1 = 330 + 60 * pre_row;
                    int pre_x2 = pre_x1 + 185;
                    int pre_y2 = pre_y1 + 50;

                    Fill_Rounded_Rectangle(pre_x1, pre_y1, pre_x2,
pre_y2, 5, white);
                    Draw_Rounded_Rectangle(pre_x1, pre_y1, pre_x2,
pre_y2, 5, 1, deepblue);
                                PrintCC(pre_x1 + 17, pre_y1 + 13,
canteen[*last_index].name, HEI, 24, 1, deepblue);
                }

                // 当前按钮高亮
                Fill_Rounded_Rectangle(x1, y1, x2, y2, 5, deepblue);
                    Draw_Rounded_Rectangle(x1, y1, x2, y2, 5, 1,
deepblue);
                PrintCC(x1 + 17, y1 + 13, canteen[index].name, HEI,
24, 1, white);

                *last_index = index;
                return index + 1; // 返回食堂编号（1~18）
            }
        }
```

```
        }
    }
/*****************************************************************/
       #include "all_func.h"
    //屏幕宽度 1024，高度 768
    USER users={0};//存储信息的用户结构体

    void user_register(){

        char judge[12]="\0";//用于判断的密码

        users.state=-1;//初始化为-1，代表未绑定

        mouse_off_arrow(&mouse);

        draw_register();

        mouse_on_arrow(mouse);

        while(1){
            mouse_show_arrow(&mouse);

            if(mouse_press(300,490, 480, 540)==1){
                return;
                //welcome();//首页
            }
            else if(mouse_press(300, 250, 420, 300)==1)
            {
                press(1);//按下"用户"
                users.type=1;
            }
            else if(mouse_press(300, 330, 420, 380)==1)
            {
```

```
            press(2);//按下"商家"
            users.type=2;
        }
        else if(mouse_press(300, 410, 420, 460)==1)
        {
            press(3);//按下"骑手"
            users.type=3;
        }
        else if(mouse_press(450, 250, 700, 300)==1)//输入账号
        {
             input_mode(users.name,users.code,judge,455,255,695,29
5,1,0);
        }
        else if(mouse_press(450, 330, 700, 380)==1)//输入密码
        {
             input_mode(users.name,users.code,judge,455,335,695,37
5,2,0);
        }
        else if(mouse_press(450, 410, 700, 460)==1)//重新输入密码
        {
             input_mode(users.name,users.code,judge,455,415,695,45
5,3,0);
        }
        if(mouse_press(520, 490, 700, 540)==1)//点击确认键
        {
            if(users.type!=0)
            {
                if(strcmp(users.name,"\0")!=0)//用户名不为空
                {
                    if(strcmp(users.code,"\0")!=0)//密码不为空
                    {
                        if(!strcmp(users.code,judge))//两次密码相同
                        {
```

```
                            if(save_user(users)==0)
                            {
                                        PrintCC(430,550,"注册成功
",HEI,24,1,lightred);
                                delay(500);
                                bar1(430,550,650,580,white);
                            }
                            else if(save_user(users)==-2)//用户名已
存在
                            {
                                    PrintCC(430,550,"用户名已被注册
",HEI,24,1,lightred);
                                delay(500);
                                bar1(430,550,650,580,white);
                            }
                        }
                        else
                        {
                                    PrintCC(430,550,"两次密码不相同
",HEI,24,1,lightred);
                            delay(500);
                            bar1(430,550,650,580,white);
                        }
                    }
                    else
                    {
                                        PrintCC(430,550,"密码为空
",HEI,24,1,lightred);
                        delay(500);
                        bar1(430,550,650,580,white);
                    }
                }
                else
```

```
                {
                                        PrintCC(430,550," 用 户 名 为 空
",HEI,24,1,lightred);
                    delay(500);
                    bar1(430,550,650,580,white);
                }
            }
            else
            {
                PrintCC(430,550,"未选择用户类型",HEI,24,1,lightred);
                delay(500);
                bar1(430,550,650,580,white);
            }
        }
    }
    }
    void draw_register()
    {
        Readbmp64k(0, 0, "bmp\\city.bmp");

        Fill_Rounded_Rectangle(250, 200, 750, 580, 30,white);//填色

        Fill_Rounded_Rectangle(450, 250, 700, 300, 5,lightgrew);//账号
栏填色
        Fill_Rounded_Rectangle(450, 330, 700, 380, 5,lightgrew);//密码
栏填色
        Fill_Rounded_Rectangle(450, 410, 700, 460, 5,lightgrew);//确认
密码栏填色

        Draw_Rounded_Rectangle(300, 250, 420, 300, 25, 1,0x0235);//用
户按钮
        Draw_Rounded_Rectangle(300, 330, 420, 380, 25, 1,0x0235);//商
家按钮
```

```
        Draw_Rounded_Rectangle(300, 410, 420, 460, 25, 1,0x0235);//骑
手按钮

        Fill_Rounded_Rectangle(300,490, 480, 540, 5,0x435c);//返回按钮
        Draw_Rounded_Rectangle(520, 490, 700, 540, 5,1,0x0235);//立即
注册按钮

        PrintCC(455,265,"账号",HEI,24,1,0XC618);
        PrintCC(455,345,"密码",HEI,24,1,0XC618);
        PrintCC(455,425,"确认密码",HEI,24,1,0XC618);
        PrintCC(360,503,"返回",HEI,24,1,0xFFFF);
        PrintCC(560,503,"立即注册",HEI,24,1,0x0235);
        PrintCC(335,265,"用户",HEI,24,1,0x0235);
        PrintCC(335,345,"商家",HEI,24,1,0x0235);
        PrintCC(335,425,"骑手",HEI,24,1,0x0235);
    }

    void press(int x){
        mouse_off_arrow(&mouse);
        switch (x)
        {
            case 1:{
                Fill_Rounded_Rectangle(300, 250, 420, 300, 25,0x0235);
                    Draw_Rounded_Rectangle(300,  250,  420,  300,  25,
1,0x0235);
                PrintCC(335,265,"用户",HEI,24,1,0xFFFF);
                Fill_Rounded_Rectangle(300, 330, 420, 380, 25,0xFFFF);
                    Draw_Rounded_Rectangle(300,  330,  420,  380,  25,
1,0x0235);
                PrintCC(335,345,"商家",HEI,24,1,0x0235);
                Fill_Rounded_Rectangle(300, 410, 420, 460, 25,0xFFFF);
                    Draw_Rounded_Rectangle(300,  410,  420,  460,  25,
1,0x0235);
```

```
                PrintCC(335,425,"骑手",HEI,24,1,0x0235);
                break;
            }
            case 2:{
                Fill_Rounded_Rectangle(300, 250, 420, 300, 25,0xFFFF);
                    Draw_Rounded_Rectangle(300,  250,  420,  300,  25,
1,0x0235);
                PrintCC(335,265,"用户",HEI,24,1,0x0235);
                Fill_Rounded_Rectangle(300, 330, 420, 380, 25,0x0235);
                    Draw_Rounded_Rectangle(300,  330,  420,  380,  25,
1,0x0235);
                PrintCC(335,345,"商家",HEI,24,1,0xFFFF);
                Fill_Rounded_Rectangle(300, 410, 420, 460, 25,0xFFFF);
                    Draw_Rounded_Rectangle(300,  410,  420,  460,  25,
1,0x0235);
                PrintCC(335,425,"骑手",HEI,24,1,0x0235);
                break;
            }
            case 3:{
                Fill_Rounded_Rectangle(300, 250, 420, 300, 25,0xFFFF);
                    Draw_Rounded_Rectangle(300,  250,  420,  300,  25,
1,0x0235);
                PrintCC(335,265,"用户",HEI,24,1,0x0235);
                Fill_Rounded_Rectangle(300, 330, 420, 380, 25,0xFFFF);
                    Draw_Rounded_Rectangle(300,  330,  420,  380,  25,
1,0x0235);
                PrintCC(335,345,"商家",HEI,24,1,0x0235);
                Fill_Rounded_Rectangle(300, 410, 420, 460, 25,0x0235);
                    Draw_Rounded_Rectangle(300,  410,  420,  460,  25,
1,0x0235);
                PrintCC(335,425,"骑手",HEI,24,1,0xFFFF);
                break;
            }
```

```
        }
        mouse_on_arrow(mouse);
    }
/*****************************************************************/
    #include "all_func.h"

    #define ORDERS_PER_PAGE   4
    #define ORDER_SUPERMARKET 0
    #define ORDER_FOOD        1
    #define ORDER_DELIVER     2

    void draw_order_detail_header(int type, int local_index,OrderList
*OL, FoodList *FL, DeliverList *DL)
    {
        char buf[128];
        // 通用：下单时间、手机号
        char time_str[64];
        char phone_str[64];
        char show_distance[30];
        char show_deliver_price[30];
        int distance_m;
        float distance_km;
        float item_price;
        float deliver_price;
        if (type == ORDER_SUPERMARKET)
        {
            Order *o = &OL->elem[local_index];
            sprintf(time_str, "下单时间：%s", o->order_time);
            sprintf(phone_str, "手机号：%s", o->user_phone);
            // 取货点
            sprintf(buf, "取货点：%s", node[o->pick_up_location].name);
            PrintText(200, 150 + 150, buf, HEI, 24, 1, black);
            // 用户地址
```

```
            sprintf(buf, "用户地址：%s", node[o->destination].name);
            PrintText(500, 150 + 150, buf, HEI, 24, 1, black);
            // 距离
                  distance_m  =  dijkstra(&node[o->pick_up_location],
&node[o->destination],3); // 计算距离
            distance_km = distance_m / 1000.0; // 转换为公里
            sprintf(show_distance, "距离：%.2fkm", distance_km);
              PrintText(200, 200 + 150, show_distance, HEI, 24, 1,
0x0000);

            item_price = o->total_amount; // 获取商品价格
                   deliver_price  =  rider_deliver_price(distance_m,
item_price); // 计算配送费用
                  sprintf(show_deliver_price, "配送费：%.1f 元",
deliver_price);
             PrintText(500, 200+150, show_deliver_price, HEI, 24, 1,
0x0000);

        }
        else if (type == ORDER_FOOD)
        {
            FoodOrder *f = &FL->elem[local_index];
            sprintf(time_str, "下单时间：%s", f->order_time);
            sprintf(phone_str, "手机号：%s", f->user_phone);
            // 取餐点
            sprintf(buf, "取货点：%s", node[f->station].name);
            PrintText(200, 150 + 150, buf, HEI, 24, 1, black);
            // 用户地址
            sprintf(buf, "用户地址：%s", node[f->destination].name);
            PrintText(500, 150 + 150, buf, HEI, 24, 1, black);
            //距离
                        distance_m   =   dijkstra(&node[f->station],
&node[f->destination],3); // 计算距离
```

```
            distance_km = distance_m / 1000.0; // 转换为公里
            sprintf(show_distance, "距离：%.2fkm", distance_km);
              PrintText(200, 200 + 150, show_distance, HEI, 24, 1,
0x0000);

            item_price = f->total_amount; // 获取商品价格
                  deliver_price = rider_deliver_price(distance_m,
item_price); // 计算配送费用
                sprintf(show_deliver_price, "配送费：%.1f 元",
deliver_price);
             PrintText(500, 200+150, show_deliver_price, HEI, 24, 1,
0x0000);
        }
        else if (type == ORDER_DELIVER)
        {
            Deliver *d = &DL->elem[local_index];
            sprintf(time_str, "下单时间：%s", d->time);
            sprintf(phone_str, "手机号：%s", d->number);
            // 取货点
            sprintf(buf, "取货点：%s", node[d->station].name);
            PrintText(200, 150 + 150, buf, HEI, 24, 1, black);
            // 用户地址
            sprintf(buf, "用户地址：%s", node[d->index].name);
            PrintText(500, 150 + 150, buf, HEI, 24, 1, black);
            //距离
                        distance_m = dijkstra(&node[d->station],
&node[d->index],3); // 计算距离
            distance_km = distance_m / 1000.0; // 转换为公里
            sprintf(show_distance, "距离：%.2fkm", distance_km);
              PrintText(200, 200 + 150, show_distance, HEI, 24, 1,
0x0000);

            item_price = 2.0; // 获取商品价格
```

```
                deliver_price   =   rider_deliver_price(distance_m,
item_price); // 计算配送费用
                sprintf(show_deliver_price,  "配送费：%.1f 元",
deliver_price);
            PrintText(500, 200+150, show_deliver_price, HEI, 24, 1,
0x0000);
        }
        // 打印通用字段（相对偏移）
        PrintText(200, 50  + 150, time_str, HEI, 24, 1, black);
        PrintText(200, 100 + 150, phone_str, HEI, 24, 1, black);
    }

    void  draw_order_detail(int  type,OrderList  *OL,  FoodList  *FL,
DeliverList *DL,
        int local_index, int page)
    {
        int i;
        Order currentOrder ;
        FoodOrder currentFood;

        char current_time[20]; // 获取当前时间
        char time_str[100]; // 打印下单时间
        char user_name[100]; // 打印用户名
        char user_phone[100]; // 打印用户手机号
        char order_number; // 打印订单号
        char address[100]; // 打印用户地址
        int startIdx = 0;// 起始商品索引
        int itemsPerPage = 0;// 每页商品数量
        int endIdx = 0;// 结束商品索引
        int item_y = 0;// 商品框的 y 坐标

        float total_amount = 0.0; // 总金额
        char total_str[20]; // 总金额字符串
```

```
        int fullPageItemCount = 0; // 满页商品数量

         bar1(0, 150, 1024, 768, white);// 将详情页绘制在 0,150 到
1024,768 区域

        currentOrder = OL->elem[local_index]; // 当前订单
        currentFood = FL->elem[local_index]; // 当前订单

        // 分页按钮（超市和外卖类型）
        if (type != ORDER_DELIVER) {
                Draw_Rounded_Rectangle(220, 700, 340, 750, 25, 1,
deepblue);
            PrintCC(245, 715, "上一页", HEI, 24, 1, deepblue);
                Draw_Rounded_Rectangle(420, 700, 540, 750, 25, 1,
deepblue);
            PrintCC(445, 715, "下一页", HEI, 24, 1, deepblue);
        }

        // 接单按钮
        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5, 1, deepblue);
        PrintCC(850, 715, "接单", HEI, 24, 1, deepblue);


        // 第一页绘制头部
        if(page==0&&type!=ORDER_DELIVER)
        {
            draw_order_detail_header(type, local_index, OL, FL, DL);

            // 表头
            PrintCC(200, 400, "商品详情", HEI, 24, 1, black);
            PrintCC(650, 400, "数量", HEI, 24, 1, black);
            PrintCC(800, 400, "金额", HEI, 24, 1, black);
            PrintText(200, 420, "------------------------------", HEI,
```

```
32, 1, black);

            startIdx = 0;
            itemsPerPage = 4;//第一页 4 个，第二页后面 9 个
        }
        else if(page==0&&type==ORDER_DELIVER)
        {
            draw_order_detail_header(type, local_index, OL, FL, DL);
        }
        else // 其他页
        {
            startIdx = 4 + (page - 1) * 9;
            itemsPerPage = 9;
        }

        endIdx = startIdx + itemsPerPage;

        if (type == ORDER_DELIVER) {
            // 只展示取件码
            char code_buf[64];
            Deliver *d = &DL->elem[local_index];
            sprintf(code_buf, "取件码：%s", d->code);
            PrintText(200, 400, code_buf, HEI, 32, 1, black);
        } else {


            if (type == ORDER_SUPERMARKET)//超市订单
            {
                if (endIdx > currentOrder.itemCount)// 防止越界
                    endIdx = currentOrder.itemCount;
            }
            else if(type==ORDER_FOOD)//食堂订单
            {
```

```
            if (endIdx > currentFood.itemCount)// 防止越界
                endIdx = currentFood.itemCount;
        }

        item_y = (page == 0) ? 450 : 200;
        for (i = startIdx; i < endIdx; i++) {
            char total_str[100]; // 商品总价
            char quantity_str[100]; // 商品数量

            if(type == ORDER_SUPERMARKET)//超市订单
            {
                 int quantity = currentOrder.item[i].quantity; //
商品数量
                 float price = currentOrder.item[i].price; // 商品
价格

                sprintf(total_str, "%.2f", price * quantity);
                sprintf(quantity_str, "x%d", quantity);

                  PrintCC(200, item_y, currentOrder.item[i].name,
HEI, 24, 1, black); // 商品名
            }
            else if(type==ORDER_FOOD)//食堂订单
            {
                int quantity = currentFood.item[i].quantity; // 商
品数量
                float price = currentFood.item[i].price; // 商品价
格

                sprintf(total_str, "%.2f", price * quantity);
                sprintf(quantity_str, "x%d", quantity);

                   PrintCC(200, item_y, currentFood.item[i].name,
```

```
HEI, 24, 1, black); // 商品名
                }

                 PrintText(650, item_y, (unsigned char*)quantity_str,
HEI, 24, 1, black);// 商品数量
                PrintText(800, item_y, (unsigned char*)total_str, HEI,
24, 1, black);// 商品总价

                item_y += 50;
            }

            // 判断是否需要在此页显示总金额（当前页没有满）
            fullPageItemCount = (page == 0) ? 4 : 9;// 第一页显示 4 个商
品，其余页显示 9 个商品

            if (type == ORDER_SUPERMARKET)
            {
                                if    ((endIdx    -    startIdx)    <
fullPageItemCount||endIdx==currentOrder.itemCount) {// 当前页商品数量
不满一页或最后一个商品刚好满页都要打印出总金额
                    //如果不是最后一个商品但是满页就不打印总金额
                    // 打印分隔线
                     PrintText(200, item_y - 30, "-------------------
-----------", HEI, 32, 1, black);

                    // 计算总金额
                    total_amount = 0.0;
                    for (i = 0; i < currentOrder.itemCount; i++) {
                         int quantity = currentOrder.item[i].quantity;
// 商品数量
                          float price = currentOrder.item[i].price; //
商品价格
                        total_amount += price * quantity;
```

```
                }

                sprintf(total_str, "总金额:%.2f 元", total_amount);
                PrintText(650, item_y + 10, total_str, HEI, 24, 1,
black);

            }
        }
        else
        {
                            if    ((endIdx    -    startIdx)    <
fullPageItemCount||endIdx==currentFood.itemCount) {// 当前页商品数量不
满一页或最后一个商品刚好满页都要打印出总金额
                //如果不是最后一个商品但是满页就不打印总金额
                // 打印分隔线
                 PrintText(200, item_y - 30, "----------------------
-----------", HEI, 32, 1, black);

                // 计算总金额
                total_amount = 0.0;
                for (i = 0; i < currentFood.itemCount; i++) {
                     int quantity = currentFood.item[i].quantity;
// 商品数量
                    float price = currentFood.item[i].price; // 商
品价格
                    total_amount += price * quantity;
                }

                sprintf(total_str, "总金额:%.2f 元", total_amount);
                PrintText(750, item_y + 10, total_str, HEI, 24, 1,
black);

            }
```

```
            }
        }
    }

    //进入所接订单详情
    void show_order_detail(int local_index, int type, int user_pos)
    {
        OrderList OL = {0};
        FoodList FL = {0};
        DeliverList DL = {0};
        int page = 0;
        int totalPage = 1;

        ReadAllOrder(&OL);
        ReadAllFood(&FL);
        ReadAllDeliver(&DL);

        // 计算总页数//第一页 4 个，第二页后面 9 个
        if (type == ORDER_SUPERMARKET) {
            totalPage = (OL.elem[local_index].itemCount - 4 + 8) / 9
+ 1;
        } else if (type == ORDER_FOOD) {
            totalPage = (FL.elem[local_index].itemCount - 4 + 8) / 9
+ 1;
        }
        // ORDER_DELIVER 保持 totalPage=0

        // 绘制订单详情
        draw_order_detail(type, &OL, &FL, &DL, local_index, page);

        mouse_on_arrow(mouse);
        while (1) {
            mouse_show_arrow(&mouse);
```

```
            // 返回按钮（假设位置同原来）
            if(mouse_press(122, 50, 242, 100)==1) //返回
            {
                DestroyOList(&OL); // 释放订单列表内存
                DestroyFList(&FL); // 释放食品列表内存
                DestroyDList(&DL); // 释放快递列表内存
                return;
                //business(users.pos);
            }
            else if(mouse_press(562, 50, 682, 100)==1) //路线
            {
                press3(2); //按钮高亮
                mouse_off_arrow(&mouse);
                bar1(0, 150, 1024, 768, white); // 清除接单界面残留
                DestroyOList(&OL); // 释放订单列表内存
                DestroyFList(&FL); // 释放食品列表内存
                DestroyDList(&DL); // 释放快递列表内存
                 route(cur_orders,num_of_orders.cur_count,user_pos);//
骑手路线规划
                //return 后从这开始
                ReadAllOrder(&OL);
                ReadAllFood(&FL);
                ReadAllDeliver(&DL);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); // 清除路线界面残留
                 draw_accept_order(page,&OL,&FL,&DL); // 重新绘制订单列
表
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(782, 50, 902, 100)==1) //我的
            {
                press3(3); //按钮高亮
                press4(1);
```

```
                DestroyOList(&OL); // 释放订单列表内存
                DestroyFList(&FL); // 释放食品列表内存
                DestroyDList(&DL); // 释放快递列表内存
                my_information(user_pos);//进入我的信息界面
                //return 后从这开始
                ReadAllOrder(&OL);
                ReadAllFood(&FL);
                ReadAllDeliver(&DL);
                mouse_off_arrow(&mouse);
                bar1(0, 150, 1024, 768, white);
                draw_rider();
                mouse_on_arrow(mouse);
            }
            else if (mouse_press(220, 700, 340, 750) == 1)
            {
                if (page > 0) {
                    page--;
                    draw_order_detail(type, &OL, &FL, &DL, local_index,
page);
                } else {
                    // 提示：已是第一页
                         PrintCC(630, 715, "已是第一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(630, 715, 780, 765, white);
                }
            }
            else if (mouse_press(420, 700, 540, 750) == 1)
            {
                if (page < totalPage - 1) {
                    page++;
                    draw_order_detail(type, &OL, &FL, &DL, local_index,
page);
```

```
                } else {
                    // 提示：已是最后一页
                        PrintCC(630, 715, "已是最后一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(630, 715, 780, 765, white);
                }
            }
            else if(mouse_press(800, 700, 1000, 750) == 1)  //点击接单
            {
                if (num_of_orders.cur_count == 4)
                {
                    //打印提示
                     PrintText(100, 100, "接单数量已达上限！", HEI, 24,
1, Red);
                    delay(500);
                    bar1(100,100,500,130,deepblue);
                }
                else
                {
                     add_my_accept(&OL, &FL, &DL, type, local_index);
//加入接单列表
                    //重画订单展示列表
                    bar1(0, 150, 1024, 768, white);
                    draw_accept_order(page, &OL, &FL, &DL);
                }
            }
        }
    }

/*****************************************************************/
    #include  <all_func.h>
```

```
#define ORDERS_PER_PAGE   4
#define ORDER_SUPERMARKET 0
#define ORDER_FOOD        1
#define ORDER_DELIVER     2

void accept_order(int user_pos) // 接单界面
{
    int page = 0; // 当前页码
    int clicked;
    int type=0, local_index=0, global_index=0; //type 为订单种类，超市为 0，外卖为 1，代取为 2；local_index 为各订单在当前种类中的序号；global_index 为各订单在所有订单中的序号
    //char debg[20];
    OrderList OL = {0}; //创建超市，外卖，订单线性空表，便于读取
    FoodList FL = {0};
    DeliverList DL = {0};
    ReadAllOrder(&OL); // 读取订单列表
    ReadAllFood(&FL); // 读取食品列表
    ReadAllDeliver(&DL); // 读取快递列表
    draw_accept_order(page,&OL,&FL,&DL); // 绘制接单页面
    mouse_off_arrow(&mouse);
    //press3(1);
    num_of_orders.total_cnt = OL.length + FL.length + DL.length; // 计算订单总数
    mouse_on_arrow(mouse);
    //sprintf(debg,"%d",num_of_orders.cur_count);
    //PrintText(1,1,debg,HEI,32,1,black);
    while(1)
    {
        mouse_show_arrow(&mouse);

        if(mouse_press(122, 50, 242, 100)==1) //返回
        {
```

```
                DestroyOList(&OL); // 释放订单列表内存
                DestroyFList(&FL); // 释放食品列表内存
                DestroyDList(&DL); // 释放快递列表内存
                return;
                //business(users.pos);
            }
            else if(mouse_press(562, 50, 682, 100)==1) //路线
            {
                press3(2); //按钮高亮
                mouse_off_arrow(&mouse);
                bar1(0, 150, 1024, 768, white); // 清除接单界面残留
                DestroyOList(&OL); // 释放订单列表内存
                DestroyFList(&FL); // 释放食品列表内存
                DestroyDList(&DL); // 释放快递列表内存
                  route(cur_orders,num_of_orders.cur_count,user_pos);
//进入路线界面
                //return 后从这开始
                press3(1); //按钮高亮
                mouse_on_arrow(mouse);
                ReadAllOrder(&OL); // 读取订单列表
                ReadAllFood(&FL); // 读取食品列表
                ReadAllDeliver(&DL); // 读取快递列表
                bar1(0, 150, 1024, 768, white); // 清除路线界面残留
                 draw_accept_order(page,&OL,&FL,&DL); // 重新绘制接单界
面
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(782, 50, 902, 100)==1) //我的
            {
                press3(3); //按钮高亮
                mouse_off_arrow(&mouse);
                my_information(user_pos);
                DestroyOList(&OL); // 释放订单列表内存
```

```
                DestroyFList(&FL); // 释放食品列表内存
                DestroyDList(&DL); // 释放快递列表内存
                //return 后从这开始
                press3(1); //按钮高亮
                ReadAllOrder(&OL); // 读取订单列表
                ReadAllFood(&FL); // 读取食品列表
                ReadAllDeliver(&DL); // 读取快递列表
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); // 清除路线界面残留
                 draw_accept_order(page,&OL,&FL,&DL); // 重新绘制接单界
面
                mouse_on_arrow(mouse);
            }
            else if (mouse_press(220, 700, 340, 750) == 1) // 上一页
            {
                if (page > 0) //判断是否能上一页
                {
                    page--;
                     draw_accept_order(page,&OL,&FL,&DL); // 绘制用户订
单页面
                }
                else // 打印提示
                {
                        PrintCC(550, 700, "已是第一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(550, 700, 700, 750, white); //覆盖打印字样
                }
            }
            else if (mouse_press(420, 700, 540, 750) == 1)  // 下一页
            {
                 if ((page + 1) * 4 < num_of_orders.total_cnt)  //判断
能否下一页
```

```
                {
                    page++;
                     draw_accept_order(page,&OL,&FL,&DL); // 绘制用户订
单页面
                }
                else // 打印提示
                {
                        PrintCC(550, 700, "已是最后一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(550, 700, 700, 750, white);
                }
            }
             else if (mouse_press(750, 170 + 25, 850, 170 + 75) == 1)
//点击页面第一个订单详情按钮的位置
            {
                 global_index = page * ORDERS_PER_PAGE + 0; //计算全局
索引
                 get_ordtyp_locind(global_index,&type,&local_index,&OL
,&FL,&DL); //由全局索引计算订单类型和内部索引
                if (global_index < num_of_orders.total_cnt) //防止越界
                {
                    DestroyOList(&OL); // 释放订单列表内存
                    DestroyFList(&FL); // 释放食品列表内存
                    DestroyDList(&DL); // 释放快递列表内存
                    show_order_detail(local_index, type, user_pos); //
进入订单详情页
                    //return 后从这返回
                    press3(1); //按钮高亮
                    ReadAllOrder(&OL); // 读取订单列表
                    ReadAllFood(&FL); // 读取食品列表
                    ReadAllDeliver(&DL); // 读取快递列表
                    bar1(0,150,1024,768,white); //清楚订单详情页残留
```

```
                    draw_accept_order(page,&OL,&FL,&DL); //画订单展示
页
                }
            }
            else if (mouse_press(750, 290 + 25, 850, 290 + 75) == 1)
//点击页面第二个订单详情按钮的位置
            {
                global_index = page * ORDERS_PER_PAGE + 1;
                get_ordtyp_locind(global_index,&type,&local_index,&OL
,&FL,&DL);
                if (global_index < num_of_orders.total_cnt)
                {
                    DestroyOList(&OL); // 释放订单列表内存
                    DestroyFList(&FL); // 释放食品列表内存
                    DestroyDList(&DL); // 释放快递列表内存
                    show_order_detail(local_index, type, user_pos);
                    //return 后从这返回
                    press3(1); //按钮高亮
                    ReadAllOrder(&OL); // 读取订单列表
                    ReadAllFood(&FL); // 读取食品列表
                    ReadAllDeliver(&DL); // 读取快递列表
                    bar1(0,150,1024,768,white);
                    draw_accept_order(page,&OL,&FL,&DL);
                }
            }
            else if (mouse_press(750, 410 + 25, 850, 410 + 75) == 1)
//点击页面第三个订单详情按钮的位置
            {
                global_index = page * ORDERS_PER_PAGE + 2;
                get_ordtyp_locind(global_index,&type,&local_index,&OL
,&FL,&DL);
                if (global_index < num_of_orders.total_cnt)
                {
```

```
                DestroyOList(&OL); // 释放订单列表内存
                DestroyFList(&FL); // 释放食品列表内存
                DestroyDList(&DL); // 释放快递列表内存
                show_order_detail(local_index, type, user_pos);
                //return 后从这返回
                press3(1); //按钮高亮
                ReadAllOrder(&OL); // 读取订单列表
                ReadAllFood(&FL); // 读取食品列表
                ReadAllDeliver(&DL); // 读取快递列表
                bar1(0,150,1024,768,white);
                draw_accept_order(page,&OL,&FL,&DL);
            }
        }
        else if (mouse_press(750, 530 + 25, 850, 530 + 75) == 1)
//点击页面第四个订单详情按钮的位置
        {
            global_index = page * ORDERS_PER_PAGE + 3;
            get_ordtyp_locind(global_index,&type,&local_index,&OL
,&FL,&DL);
            if (global_index < num_of_orders.total_cnt)
            {
                DestroyOList(&OL); // 释放订单列表内存
                DestroyFList(&FL); // 释放食品列表内存
                DestroyDList(&DL); // 释放快递列表内存
                show_order_detail(local_index, type, user_pos);
                //return 后从这返回
                press3(1); //按钮高亮
                ReadAllOrder(&OL); // 读取订单列表
                ReadAllFood(&FL); // 读取食品列表
                ReadAllDeliver(&DL); // 读取快递列表
                bar1(0,150,1024,768,white);
                draw_accept_order(page,&OL,&FL,&DL);
            }
```

```
            }
            else if (mouse_press(875, 170 + 25, 975, 170 + 75) == 1)
//点击页面第一个订单接单按钮的位置
            {
                global_index = page * ORDERS_PER_PAGE + 0;
                get_ordtyp_locind(global_index,&type,&local_index,&OL
,&FL,&DL);

                if (num_of_orders.cur_count == 4)
                {
                    //打印提示
                     PrintText(100, 100, "接单数量已达上限！", HEI, 24,
1, Red);
                    delay(500);
                    bar1(100,100,500,130,deepblue);
                }
                else
                {
                     add_my_accept(&OL, &FL, &DL, type, local_index);
//加入接单列表
                    //重画订单展示列表
                    bar1(0, 150, 1024, 768, white);
                    draw_accept_order(page, &OL, &FL, &DL);
                }
            }
            else if (mouse_press(875, 290 + 25, 975, 290 + 75) == 1)
//点击页面第二个订单接单按钮的位置
            {
                global_index = page * ORDERS_PER_PAGE + 1;
                get_ordtyp_locind(global_index,&type,&local_index,&OL
,&FL,&DL);
                if (global_index < num_of_orders.total_cnt)
                {
```

```
                if ( num_of_orders.cur_count== 4)
                {
                     PrintText(100, 100, "接单数量已达上限！", HEI,
24, 1, Red);
                    delay(500);
                    bar1(100,100,500,130,deepblue);
                }
                else
                {
                            add_my_accept(&OL,  &FL,  &DL,  type,
local_index);
                    //重画订单展示列表
                    bar1(0, 150, 1024, 768, white);
                    draw_accept_order(page, &OL, &FL, &DL);
                }
            }
        }
        else if (mouse_press(875, 410 + 25, 975, 410 + 75) == 1)
//点击页面第三个订单接单按钮的位置
        {
            global_index = page * ORDERS_PER_PAGE + 2;
            get_ordtyp_locind(global_index,&type,&local_index,&OL
,&FL,&DL);
        if (global_index < num_of_orders.total_cnt)
        {
            if ( num_of_orders.cur_count== 4)
            {
                 PrintText(100, 100, "接单数量已达上限！", HEI, 24,
1, Red);
                delay(500);
                bar1(100,100,500,130,deepblue);
            }
            else
```

```
                {
                    add_my_accept(&OL, &FL, &DL, type, local_index);
                    bar1(0, 150, 1024, 768, white);
                    draw_accept_order(page, &OL, &FL, &DL);
                }
            }
        }
            else if (mouse_press(875, 530 + 25, 975, 530 + 75) == 1)
//点击页面第四个订单接单按钮的位置
            {
                global_index = page * ORDERS_PER_PAGE + 3;
                 get_ordtyp_locind(global_index,&type,&local_index,&OL
,&FL,&DL);
                if (global_index < num_of_orders.total_cnt)
                {
                    if ( num_of_orders.cur_count== 4)
                    {
                         PrintText(100, 100, "接单数量已达上限！", HEI,
24, 1, Red);
                        delay(500);
                        bar1(100,100,500,130,deepblue);
                    }
                    else
                    {
                                add_my_accept(&OL,  &FL,  &DL,  type,
local_index);
                        //重画订单展示列表
                        bar1(0, 150, 1024, 768, white);
                        draw_accept_order(page, &OL, &FL, &DL);
                    }

                }
            }
```

```
        }
    }

    double rider_deliver_price(int distance_m, float order_amount) //
计算配送费
    {
        const double price_per_km   = 0.5;   // 每公里配送费（元）
        const double base_price     = 2.0;   // 起步价（元）
        const double price_ratio    = 0.05;  // 商品金额 5% 加价

        double dist_km = (double)distance_m / 1000.0; //单位换算
          double  total  =  base_price  +  dist_km  *  price_per_km  +
order_amount * price_ratio;// 计算总费
        return total;
    }

    void  draw_accept_order(int  page,  OrderList  *OL,  FoodList  *FL,
DeliverList *DL) // 绘制接单页面
    {
        int i,j;
        int y_offset = 170; // 初始 Y 轴偏移
        int start_index = page * 4; // 当前页的起始订单索引
        int end_index = start_index + 4; // 当前页的结束订单索引
        char debg[20];
        if (end_index > num_of_orders.total_cnt)
        {
            end_index = num_of_orders.total_cnt; // 防止越界
        }


        bar1(0, 150, 1024, 768, white); // 清空屏幕
        //press3(1);//画按钮
        //PrintText(100,0,"111",HEI,32,1,Red);
```

```
// sprintf(debg,"总订单数：%d",num_of_orders.total_cnt);
// PrintText(0,0,debg,HEI,36,1,Red);

if(num_of_orders.total_cnt==0)
{
    bar1(0, 150, 1024, 768, white); // 清空屏幕
    PrintCC(400,400,"当前无可接订单",HEI,32,1,Red);
}

else
{
    // 绘制订单
    for (i = start_index; i < end_index; i++)
    {
        char show_pick_up[20]; // 取餐地点
        char show_destination[20]; // 目的地
        char show_distance[20]; // 距离
        char show_deliver_price[20]; // 配送费用
        int distance_m; // 距离
        float distance_km; // 距离
        float item_price;// 商品价格
        double deliver_price;//配送费

        if(i < OL->length) //先展示超市订单
        {
            //char debg[20];
            Order order = OL->elem[i]; // 获取当前订单
            //绘制订单框
            // sprintf(debg,"%d",i);
            // PrintText(0,0,debg,HEI,24,1,black);
            // sprintf(debg,"%d",order.pick_up_location);
            // PrintText(0,50,debg,HEI,32,1,black);
                  Draw_Rounded_Rectangle(20,  y_offset,  1000,
```

```
y_offset + 100, 30, 1, deepgrey);
                    //绘制详情按钮
                        Fill_Rounded_Rectangle(750, y_offset+25, 850,
y_offset+75, 25, white);
                        Draw_Rounded_Rectangle(750, y_offset+25, 850,
y_offset+75, 25, 1,deepblue);
                     PrintCC(750+25, y_offset+30, "详情", HEI, 24, 1,
deepblue);
                    //绘制接单按钮
                        Fill_Rounded_Rectangle(875, y_offset+25, 975,
y_offset+75, 25, white);
                        Draw_Rounded_Rectangle(875, y_offset+25, 975,
y_offset+75, 25, 1,deepblue);
                     PrintCC(875+25, y_offset+30, "接单", HEI, 24, 1,
deepblue);
                    // 显示订单简略信息
                              sprintf(show_pick_up, "取货点: %s",
node[order.pick_up_location].name); //展示取货点
                     PrintText(50, y_offset + 10, show_pick_up, HEI,
24, 1, black);

                             sprintf(show_destination, "送货点: %s",
node[order.destination].name); //展示送货点
                    PrintText(50, y_offset + 60, show_destination, HEI,
24, 1, black);

                                                    distance_m     =
dijkstra(&node[order.pick_up_location], &node[order.destination],3);
// 计算距离
                    distance_km = distance_m / 1000.0; // 转换为公里
                    sprintf(show_distance, "距离: %.2fkm", distance_km);
//展示距离
                     PrintText(500, y_offset + 10, show_distance, HEI,
```

```
24, 1, black);

                    item_price = order.total_amount; // 获取商品价格
                      deliver_price = rider_deliver_price(distance_m,
item_price); // 计算配送费用
                        sprintf(show_deliver_price, "配送费：%.1f 元",
deliver_price);
                    PrintText(500, y_offset + 60, show_deliver_price,
HEI, 24, 1, black);

                    y_offset += 120; // 每个订单框之间的间距
                }
                else if (i >= OL->length && i < OL->length + FL->length)
// 然后展示食堂订单
                {

                     FoodOrder food_order = FL->elem[i - OL->length];
// 获取当前订单

                        Draw_Rounded_Rectangle(20, y_offset, 1000,
y_offset + 100, 30, 1, 0x6B4D);

                      Fill_Rounded_Rectangle(750, y_offset+25, 850,
y_offset+75, 25, white);
                      Draw_Rounded_Rectangle(750, y_offset+25, 850,
y_offset+75, 25, 1,deepblue);
                     PrintCC(750+25, y_offset+30, "详情", HEI, 24, 1,
deepblue);

                      Fill_Rounded_Rectangle(875, y_offset+25, 975,
y_offset+75, 25, white);
                      Draw_Rounded_Rectangle(875, y_offset+25, 975,
y_offset+75, 25, 1,deepblue);
```

```
                    PrintCC(875+25, y_offset+30, "接单", HEI, 24, 1,
deepblue);

                              sprintf(show_pick_up, "取餐点：%s",
node[food_order.station].name);
                     PrintText(50, y_offset + 10, show_pick_up, HEI,
24, 1, black);

                             sprintf(show_destination, "送餐点：%s",
node[food_order.destination].name);
                   PrintText(50, y_offset + 60, show_destination, HEI,
24, 1, black);

                    distance_m = dijkstra(&node[food_order.station],
&node[food_order.destination],3);
                   distance_km = distance_m / 1000.0;
                   sprintf(show_distance, "距离：%.2fkm", distance_km);
                    PrintText(500, y_offset + 10, show_distance, HEI,
24, 1, black);

                   item_price = food_order.total_amount;
                     deliver_price = rider_deliver_price(distance_m,
item_price);
                        sprintf(show_deliver_price, "配送费：%.1f元",
deliver_price);
                    PrintText(500, y_offset + 60, show_deliver_price,
HEI, 24, 1, black);

                   y_offset += 120; // 每个订单框之间的间距
               }
               else if(i >= OL->length + FL->length)//最后展示快递代取
订单
               {
```

```
                    Deliver deliver = DL->elem[i - OL->length -
FL->length];

                      Draw_Rounded_Rectangle(20, y_offset, 1000,
y_offset + 100, 30, 1, 0x6B4D);

                    Fill_Rounded_Rectangle(750, y_offset+25, 850,
y_offset+75, 25, white);
                    Draw_Rounded_Rectangle(750, y_offset+25, 850,
y_offset+75, 25, 1,deepblue);
                  PrintCC(750+25, y_offset+30, "详情", HEI, 24, 1,
deepblue);

                    Fill_Rounded_Rectangle(875, y_offset+25, 975,
y_offset+75, 25, white);
                    Draw_Rounded_Rectangle(875, y_offset+25, 975,
y_offset+75, 25, 1,deepblue);
                  PrintCC(875+25, y_offset+30, "接单", HEI, 24, 1,
deepblue);

                         sprintf(show_pick_up, "取货点：%s",
node[deliver.station+408].name);
                   PrintText(50, y_offset + 10, show_pick_up, HEI,
24, 1, black);

                       sprintf(show_destination, "送货点：%s",
node[deliver.index].name);
                 PrintText(50, y_offset + 60, show_destination, HEI,
24, 1, black);

                    distance_m = dijkstra(&node[deliver.station],
&node[deliver.index],3);
                 distance_km = distance_m / 1000.0;
```

```
                    sprintf(show_distance, "距离：%.2fkm", distance_km);
                    PrintText(500, y_offset + 10, show_distance, HEI,
24, 1, black);

                    item_price = 2.0;
                      deliver_price = rider_deliver_price(distance_m,
item_price);
                         sprintf(show_deliver_price, "配送费：%.1f 元",
deliver_price);
                    PrintText(500, y_offset + 60, show_deliver_price,
HEI, 24, 1, black);

                    y_offset += 120;
                }
            }
        }

        // 绘制翻页按钮
        Draw_Rounded_Rectangle(220, 700, 340, 750, 25, 1, deepblue);
// 上一页
        Draw_Rounded_Rectangle(420, 700, 540, 750, 25, 1, deepblue);
// 下一页
        PrintCC(245, 715, "上一页", HEI, 24, 1, deepblue);
        PrintCC(445, 715, "下一页", HEI, 24, 1, deepblue);
    }

    void get_ordtyp_locind(int global_index,
        int *type, int *local_index,
        const OrderList *OL, const FoodList *FL, const DeliverList *DL)
//基于先显示超市订单，再显示外卖订单，最后显示代取订单，由总索引计算订单类型
和本地索引
    {
        if (global_index < OL->length) //全局索引小于超市订单数，则为超
```

```
市订单
        {
            *type = ORDER_SUPERMARKET;
            *local_index = global_index;
        }
        else if (global_index < OL->length + FL->length) //全局索引小于超市加外卖订单，则为外卖订单
        {
            *type = ORDER_FOOD;
            *local_index = global_index - OL->length; //本地索引由总索引减去超市订单数
        }
        else if (global_index < OL->length + FL->length + DL->length) //剩余的为代取订单
        {
            *type = ORDER_DELIVER;
            *local_index = global_index - OL->length - FL->length; //本地索引由总索引减去超市订单数和外卖订单数
        }
    }

/*****************************************************************/
    #include <all_func.h>
    #define MAX_COMBINED_ORDERS 20
    #define ORDERS_PER_PAGE   4

    #define ORDER_SUPERMARKET 0
    #define ORDER_FOOD        1
    #define ORDER_DELIVER     2

    int  arrange(int  start_idx,  AcceptedOrder  acp_orders[],  int n_orders)
    {
```

```
int temp_picked[4]; // 临时数组，用于存储当前订单的取餐状态
int temp_delivered[4];  // 临时数组，用于存储当前订单的送餐状态
int temp_remaining;
int temp_current;
int temp_step;
int best_i;
int best_type;    // 0 = 取餐, 1 = 送餐
int dist;
int best_dist;
int i;
char buf[64];
int next_index;
int next_pos;
int next_type;

char debuf[10];
for(i = 0; i < n_orders; i++)
{
    temp_picked[i] = route_state.picked[i];
    temp_delivered[i] = route_state.delivered[i];
}
temp_remaining = route_state.remaining;
temp_current = route_state.current_pos;
temp_step = 0;
/*—— 主循环 ——*/

if(temp_remaining == 0) {
    PrintText(150, 200, "您已完成所有订单", HEI, 24, 1, black);
    return -1; // 返回-1 表示没有可执行任务
}
while (temp_remaining > 0) {
    /*—— 找最近的可做任务 ——*/
    best_dist = 20000;
```

```
            best_i    = -1;
            best_type = -1;

            for (i = 0; i < n_orders; i++)
            {
                /* 取餐任务 */
                int pu_idx, dst_idx;

                if (!temp_picked[i]) {
                    // 取餐
                    if (acp_orders[i].type == ORDER_SUPERMARKET) {
                                                            pu_idx    =
acp_orders[i].data.order.pick_up_location;
                    } else if (acp_orders[i].type == ORDER_FOOD) {
                                                            pu_idx    =
acp_orders[i].data.food.pick_up_location;
                    } else {
                        pu_idx = acp_orders[i].data.deliver.station;
                    }

                    dist = Manhattan_Distance(
                        node[temp_current].x, node[temp_current].y,
                        node[pu_idx].x,        node[pu_idx].y
                    );
                    if (dist < best_dist ) {
                        best_dist = dist;
                        best_i    = i;
                        best_type = 0;
                    }
                }
                else if (!temp_delivered[i]) {
                    // 送餐
                    if (acp_orders[i].type == ORDER_SUPERMARKET) {
```

```
                    dst_idx = acp_orders[i].data.order.destination;
                } else if (acp_orders[i].type == ORDER_FOOD) {
                    dst_idx = acp_orders[i].data.food.destination;
                } else {
                    dst_idx = acp_orders[i].data.deliver.index;
                }

                dist = Manhattan_Distance(
                    node[temp_current].x, node[temp_current].y,
                    node[dst_idx].x,      node[dst_idx].y
                );
                if (dist < best_dist )
                {
                    best_dist = dist;
                    best_i    = i;
                    best_type = 1;
                }
            }
        }

        temp_step++;
        /*—— 如果没有找到任务，就跳出 ——*/\
        if(temp_step==1)
        {
            if(best_type == 0)
            {
                 if (acp_orders[best_i].type == ORDER_SUPERMARKET)
{
                                                        next_pos    =
acp_orders[best_i].data.order.pick_up_location;
                 } else if (acp_orders[best_i].type == ORDER_FOOD)
{
                                                        next_pos    =
```

```
acp_orders[best_i].data.food.pick_up_location;
                    } else {
                                                                next_pos    =
acp_orders[best_i].data.deliver.station;
                    }
                    next_type = 0;
                }
                else
                {
                     if (acp_orders[best_i].type == ORDER_SUPERMARKET)
{
                                                                next_pos    =
acp_orders[best_i].data.order.destination;
                     } else if (acp_orders[best_i].type == ORDER_FOOD)
{
                                                                next_pos    =
acp_orders[best_i].data.food.destination;
                    } else {
                                                                next_pos    =
acp_orders[best_i].data.deliver.index;
                    }
                    next_type = 1;
                }
                next_index = best_i;
                route_state.next_pos = next_pos;
                route_state.next_type = next_type;
            }
            if (best_i < 0) {
                 PrintText(50, 50, "调度异常：无可执行任务", HEI, 24, 1,
black);
                break;
            }
```

```
            draw_arrange(temp_step, acp_orders, temp_current, best_i,
best_type);
            // 在 arrange 函数中添加位置更新逻辑
            if (best_type == 0) {
                if (acp_orders[best_i].type == ORDER_SUPERMARKET) {
                                                    temp_current     =
acp_orders[best_i].data.order.pick_up_location;
                } else if (acp_orders[best_i].type == ORDER_FOOD) {
                                                    temp_current     =
acp_orders[best_i].data.food.pick_up_location;
                } else {
                                                    temp_current     =
acp_orders[best_i].data.deliver.station;
                }
                temp_picked[best_i] = 1;
            } else {
                if (acp_orders[best_i].type == ORDER_SUPERMARKET) {
                                                    temp_current     =
acp_orders[best_i].data.order.destination;
                } else if (acp_orders[best_i].type == ORDER_FOOD) {
                                                    temp_current     =
acp_orders[best_i].data.food.destination;
                } else {
                                                    temp_current     =
acp_orders[best_i].data.deliver.index;
                }
                temp_delivered[best_i] = 1;
            }
            temp_remaining--;
        }
        return next_index;
    }
```

```
    void draw_arrange(int j, struct AcceptedOrder acp_orders[], int
start_index, int best_i, int best_type)
    {
        int text_x,text_y;
        char buf[20];
        int distance_m;
        float distance_km;
        float time_min; // 假设最小时间为 1 分钟
        int time_m;
        int time_s;
        int pu_idx, dst_idx; //取餐点和送餐点
        //Readbmp64k(0, 326, "bmp\\map4.bmp");
        Fill_Rounded_Rectangle(900, 266, 1020, 316, 5,deepblue);//已完
成
        Draw_Rounded_Rectangle(900, 266, 1020, 316, 5, 1, deepblue);//
已完成
        PrintCC(910, 276, "已完成", HEI, 24, 1, white);
        if (best_type == 0) {
            // 取餐
            switch (acp_orders[best_i].type) {
            case 0:
                                                    pu_idx       =
acp_orders[best_i].data.order.pick_up_location;
                break;
            case 1:
                pu_idx = acp_orders[best_i].data.food.station;
                break;
            case 2:
                pu_idx = acp_orders[best_i].data.deliver.station;
                break;
            }
             distance_m = dijkstra(&node[start_index], &node[pu_idx],
j);
```

```
        } else {
            // 送餐
            switch (acp_orders[best_i].type) {
            case 0:
                dst_idx = acp_orders[best_i].data.order.destination;
                break;
            case 1:
                dst_idx = acp_orders[best_i].data.food.destination;
                break;
            case 2:
                dst_idx = acp_orders[best_i].data.deliver.index;
                break;
            }
            distance_m = dijkstra(&node[start_index], &node[dst_idx],
j);
        }

        // if (best_type == 0)
               //    distance_m    =    dijkstra(&node[start_index],
&node[acp_orders[best_i].pick_up_index],j);//计算最短路径
        // else
               //    distance_m    =    dijkstra(&node[start_index],
&node[acp_orders[best_i].destination_index],j);//计算最短路径

        if(j==1) //第一个地点时打印起点
        {

                Draw_Rounded_Rectangle(10,  160,  130,  210,  5,  1,
deepblue);//起点
            sprintf(buf, "%s",node[start_index].name);
               calculate_centered_text(10,  160,  130,  210,  buf,  24,
&text_x, &text_y);
            PrintText(text_x, text_y, buf, HEI, 24, 1, black);//起点
```

```
        }

        if (j <= 4)  //地点在第一排
        {
            //画地点框
            Draw_Rounded_Rectangle(10 + 221*j, 160, 130 + 221*j, 210,
5, 1, deepblue);//1 号
            if(best_type == 0)
            {
                sprintf(buf, "%s",node[pu_idx].name);
            }
            else
            {
               sprintf(buf, "%s",node[dst_idx].name);
            }

            calculate_centered_text(10 + 221*j, 160, 130 + 221*j, 210,
buf, 24, &text_x, &text_y);
            PrintText(text_x, text_y, buf, HEI, 24, 1, black);//1 号
            //画箭头
            Line_Thick(221*j-91+3, 185, 10 + 221*j-3, 185, 3, black);//
连线
            Line_Thick(10+221*j-3, 185, 221*j-10-3, 165, 3, black);
            Line_Thick(10+221*j-3, 185, 221*j-10-3, 205, 3, black);//
箭头
            //标注距离
            distance_km = distance_m / 1000.0; // 转换为公里
            sprintf(buf, "%.2fkm", distance_km);
            calculate_centered_text(221*j-91+3, 185-16*2 , 10+221*j-3,
185 , buf, 16, &text_x, &text_y);
            PrintText(text_x, text_y, buf, HEI, 16, 1, black);//距离
            //标注时间
            time_min = distance_m / 1000.0 * 60 / 20; // 假设平均速度为
```

```
20km/h，计算时间
            if(time_min < 1.0)
            {
                time_s = (int)(time_min * 60.0 + 0.5);  // 四舍五入
                sprintf(buf, "%ds", time_s);
            }
            else
            {
                time_m = (int)(time_min + 0.5);
                sprintf(buf, "%dmin", time_m);
            }
               calculate_centered_text(221*j-91+3,  185  ,  10+221*j-3,
185+16*2 , buf, 16, &text_x, &text_y);
            PrintText(text_x, text_y, buf, HEI, 16, 1, black);//时间
        }
        else
        {
            //画地点框
            Draw_Rounded_Rectangle(220*j-990, 266, 220*j-870, 316, 5,
1, deepblue);//1 号
            if(best_type == 0)
            {
                // pu_idx = (acp_orders[best_i].type == 0)
                              //                                        ?
acp_orders[best_i].data.deliver.station
                 //                  : (acp_orders[best_i].type == 1)
//先判断是否是超市单，后判断是否是外卖单
                             //                                        ?
acp_orders[best_i].data.food.station
                             //                                        :
acp_orders[best_i].data.order.pick_up_location;
                sprintf(buf, "%s",node[pu_idx].name);
            }
```

```
            else
            {
                // dst_idx = (acp_orders[best_i].type == 0)
                                //                                          ?
acp_orders[best_i].data.deliver.index
                //                 : (acp_orders[best_i].type == 1)
                               //                                          ?
acp_orders[best_i].data.food.destination
                               //                                          :
acp_orders[best_i].data.order.destination;
                sprintf(buf, "%s",node[dst_idx].name);
            }
             calculate_centered_text(220*j-990, 266, 220*j-870, 316,
buf, 24, &text_x, &text_y);
            PrintText(text_x, text_y, buf, HEI, 24, 1, black);//1 号
            //画箭头
                Line_Thick(220*j-1090+3,  291,  220*j-990-3,  291,  3,
black);//连线
            Line_Thick(220*j-990-3, 291, 220*j-1010+3, 271, 3, black);
                Line_Thick(220*j-990-3,  291,  220*j-1010+3,  311,  3,
black);//箭头
            //标注距离
            sprintf(buf, "%.2fkm", distance_km);
              calculate_centered_text(220*j-1090+3 ,291-16*2 ,220*j-
990-3 ,291 , buf, 16,&text_x,&text_y);
            PrintText(text_x,text_y , buf , HEI ,16 ,1 ,black);//距离
            //标注时间
            if(time_min < 1)
            sprintf(buf, "%ds", time_s);
            else
            sprintf(buf, "%dmin", time_min);
               calculate_centered_text(220*j-1090+3  ,291  ,220*j-990-
3 ,291+16*2 , buf ,24,&text_x,&text_y);
```

```
            PrintText(text_x, text_y, buf, HEI, 16, 1, black);//时间
        }
    }

    void calculate_centered_text(int rect_x1, int rect_y1, int rect_x2,
int rect_y2, const char *text, int font_size, int *text_x, int *text_y)
    {
        int rect_width = rect_x2 - rect_x1;
        int rect_height = rect_y2 - rect_y1;

        int text_width = strlen(text) * font_size / 2;
        int text_height = font_size;


        *text_x = rect_x1 + (rect_width - text_width) / 2;
        *text_y = rect_y1 + (rect_height - text_height) / 2;
    }

    int Manhattan_Distance(int x1, int y1, int x2, int y2)
    {
        return abs(x1 - x2) + abs(y1 - y2);
    }
/*****************************************************************/
    #include "all_func.h"

    #define ORDERS_PER_PAGE   4
    #define ORDER_SUPERMARKET 0
    #define ORDER_FOOD        1
    #define ORDER_DELIVER     2

    void draw_my_order_detail_body(int type, int index)
    {
        char buf[128];
```

```
        // 通用：下单时间、手机号
        char time_str[64];
        char phone_str[64];
        char show_distance[30];
        char show_deliver_price[30];
        int distance_m;
        float distance_km;
        float item_price;
        float deliver_price;

        if (type == ORDER_SUPERMARKET)
        {
                        sprintf(time_str,   "  下  单  时  间  :  %s",
cur_orders[index].data.order.order_time);
                         sprintf(phone_str,   "  手  机  号  :  %s",
cur_orders[index].data.order.user_phone);
            // 取货点
                           sprintf(buf,    "  取  货  点  :  %s",
node[cur_orders[index].data.order.pick_up_location].name);
            PrintText(200, 150 + 150, buf, HEI, 24, 1, black);
            // 用户地址
                         sprintf(buf,    "  用  户  地  址  :  %s",
node[cur_orders[index].data.order.destination].name);
            PrintText(500, 150 + 150, buf, HEI, 24, 1, black);
            // 距离
                                                  distance_m          =
dijkstra(&node[cur_orders[index].data.order.pick_up_location],
&node[cur_orders[index].data.order.destination],3); // 计算距离
            distance_km = distance_m / 1000.0; // 转换为公里
            sprintf(show_distance, "距离：%.2fkm", distance_km);
            PrintText(200, 200 + 150, show_distance, HEI, 24, 1, black);

              item_price  =  cur_orders[index].data.order.total_amount;
```

```
// 获取商品价格
                deliver_price = rider_deliver_price(distance_m,
item_price); // 计算配送费用
            cur_orders[index].deliver_price=deliver_price; //将此单配
送费存进结构体中
                sprintf(show_deliver_price, "配送费：%.1f 元",
deliver_price);
            PrintText(500, 200+150, show_deliver_price, HEI, 24, 1,
black);

        }
        else if (type == ORDER_FOOD)
        {

                    sprintf(time_str, "下单时间：%s",
cur_orders[index].data.food.order_time);
                     sprintf(phone_str, "手机号：%s",
cur_orders[index].data.food.user_phone);
            // 取餐点
                        sprintf(buf, "取货点：%s",
node[cur_orders[index].data.food.station].name);
            PrintText(200, 150 + 150, buf, HEI, 24, 1, black);
            // 用户地址
                     sprintf(buf, "用户地址：%s",
node[cur_orders[index].data.food.destination].name);
            PrintText(500, 150 + 150, buf, HEI, 24, 1, black);
            //距离
                                            distance_m =
dijkstra(&node[cur_orders[index].data.food.station],
&node[cur_orders[index].data.food.destination],3); // 计算距离
            distance_km = distance_m / 1000.0; // 转换为公里
            sprintf(show_distance, "距离：%.2fkm", distance_km);
            PrintText(200, 200 + 150, show_distance, HEI, 24, 1, black);
```

```
            item_price = cur_orders[index].data.order.total_amount;
// 获取商品价格
                deliver_price = rider_deliver_price(distance_m,
item_price); // 计算配送费用
          cur_orders[index].deliver_price=deliver_price;
               sprintf(show_deliver_price, "配送费：%.1f 元",
deliver_price);
           PrintText(500, 200+150, show_deliver_price, HEI, 24, 1,
black);
       }
       else if (type == ORDER_DELIVER)
       {
                   sprintf(time_str, "下单时间：%s",
cur_orders[index].data.deliver.time);
                    sprintf(phone_str, "手机号：%s",
cur_orders[index].data.deliver.number);
          // 取货点
                      sprintf(buf, "取货点：%s",
node[cur_orders[index].data.deliver.station].name);
          PrintText(200, 150 + 150, buf, HEI, 24, 1, black);
          // 用户地址
                    sprintf(buf, "用户地址：%s",
node[cur_orders[index].data.deliver.destination].name);
          PrintText(500, 150 + 150, buf, HEI, 24, 1, black);
          //距离
                                              distance_m =
dijkstra(&node[cur_orders[index].data.deliver.station],
&node[cur_orders[index].data.deliver.destination],3); // 计算距离
          distance_km = distance_m / 1000.0; // 转换为公里
          sprintf(show_distance, "距离：%.2fkm", distance_km);
            PrintText(200, 200 + 150, show_distance, HEI, 24, 1,
0x0000);
```

```
            item_price = 2.0; // 获取商品价格
                   deliver_price   =   rider_deliver_price(distance_m,
item_price); // 计算配送费用
            cur_orders[index].deliver_price=deliver_price;
                 sprintf(show_deliver_price,  "配送费：%.1f 元",
deliver_price);
             PrintText(500, 200+150, show_deliver_price, HEI, 24, 1,
0x0000);
        }
        // 打印通用字段（相对偏移）
        PrintText(200, 50  + 150, time_str, HEI, 24, 1, black);
        PrintText(200, 100 + 150, phone_str, HEI, 24, 1, black);
    }

    void my_accept_detail(int index , int user_pos)
    {
        OrderList OL = {0};
        FoodList FL = {0};
        int page = 0;
        int totalPage = 1;

        int type;
        type=cur_orders[index].type;

        ReadAllOrder(&OL); // 读取订单列表
        ReadAllFood(&FL); // 读取食品列表

        // 计算总页数//第一页 4 个，第二页后面 9 个
        if (type == ORDER_SUPERMARKET) {
            totalPage = (OL.elem[index].itemCount - 4 + 8) / 9 + 1;
        } else if (type == ORDER_FOOD) {
            totalPage = (FL.elem[index].itemCount - 4 + 8) / 9 + 1;
```

```
        }
        // ORDER_DELIVER 保持 totalPage=0

        DestroyOList(&OL); // 释放快递列表内存
        DestroyFList(&FL); // 释放食品列表内存

        draw_my_order_detail(type, index,page);

        mouse_on_arrow(mouse);
        while (1)
        {
            mouse_show_arrow(&mouse);
            // 返回按钮
            if(mouse_press(122, 50, 242, 100)==1) //返回
            {
                return;
                //my_accept_order(users.pos);
            }
            else if(mouse_press(562, 50, 682, 100)==1) //路线
            {
                press3(2); //按钮高亮
                mouse_off_arrow(&mouse);
                bar1(0, 150, 1024, 768, white);
                                                    route(cur_orders,
num_of_orders.cur_count,user_pos);//骑手路线规划
                //return 后从这开始
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white);
                draw_my_order_detail(type, index,page);
            }
            else if(mouse_press(782, 50, 902, 100)==1) //我的
            {
                press3(3); //按钮高亮
```

```
                mouse_off_arrow(&mouse);
                bar1(0, 150, 1024, 768, white);
                my_information(user_pos);
                //return 后从这开始
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white);
                draw_my_order_detail(type, index,page);
            }
            else if (mouse_press(220, 700, 340, 750) == 1)
            {
                if (page > 0) {
                    page--;
                    draw_my_order_detail(type, index,page);
                } else {
                    // 提示：已是第一页
                         PrintCC(630, 715, "已是第一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(630, 715, 780, 765, white);
                }
            }
            else if (mouse_press(420, 700, 540, 750) == 1)
            {
                if (page < totalPage - 1) {
                    page++;
                    draw_my_order_detail(type, index,page);
                } else {
                    // 提示：已是最后一页
                        PrintCC(630, 715, "已是最后一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(630, 715, 780, 765, white);
                }
```

```
        }

    }
}

void draw_my_order_detail(int type,int index,int page)
{
    int i;
    Order currentOrder ;
    FoodOrder currentFood;
    Deliver currentDeliver;

    char current_time[100]; // 获取当前时间
    char time_str[100]; // 打印下单时间
    char user_name[100]; // 打印用户名
    char user_phone[100]; // 打印用户手机号
    char order_number; // 打印订单号
    char address[100]; // 打印用户地址
    int startIdx = 0;// 起始商品索引
    int itemsPerPage = 0;// 每页商品数量
    int endIdx = 0;// 结束商品索引
    int item_y = 0;// 商品框的 y 坐标

    float total_amount = 0.0; // 总金额
    char total_str[100]; // 总金额字符串
    int fullPageItemCount = 0; // 满页商品数量
    // 清屏
    bar1(0, 150, 1024, 768, white);

    currentOrder = cur_orders[index].data.order; // 当前订单
    currentFood = cur_orders[index].data.food; // 当前订单
    currentDeliver = cur_orders[index].data.deliver; // 当前快递
```

```
        // 绘制主体部分


        // 分页按钮（超市和外卖类型）
        if (type != ORDER_DELIVER) {
                Draw_Rounded_Rectangle(220,  700,  340,  750,  25,  1,
deepblue);
            PrintCC(245, 715, "上一页", HEI, 24, 1, deepblue);
                Draw_Rounded_Rectangle(420,  700,  540,  750,  25,  1,
deepblue);
            PrintCC(445, 715, "下一页", HEI, 24, 1, deepblue);
        }

        // 第一页绘制头部
        if(page==0&&type!=ORDER_DELIVER)
        {
            draw_my_order_detail_body(type, index);

            // 表头
            PrintCC(200, 400, "商品详情", HEI, 24, 1, black);
            PrintCC(650, 400, "数量", HEI, 24, 1, black);
            PrintCC(800, 400, "金额", HEI, 24, 1, black);
            PrintText(200, 420, "------------------------------", HEI,
32, 1, black);

            startIdx = 0;
            itemsPerPage = 4;//第一页 4 个，第二页后面 9 个
        }
        else if(page==0&&type==ORDER_DELIVER)
        {
            draw_my_order_detail_body(type, index);
        }
        else // 其他页
```

```
{
    startIdx = 4 + (page - 1) * 9;
    itemsPerPage = 9;
}

endIdx = startIdx + itemsPerPage;

if (type == ORDER_DELIVER) {
    // 只展示取件码
    char code_buf[64];
    sprintf(code_buf, "取件码：%s", currentDeliver.code);
    PrintText(200, 400, code_buf, HEI, 32, 1, black);
} else {


    if (type == ORDER_SUPERMARKET)//超市订单
    {
        if (endIdx > currentOrder.itemCount)// 防止越界
            endIdx = currentOrder.itemCount;
    }
    else if(type==ORDER_FOOD)//食堂订单
    {
        if (endIdx > currentFood.itemCount)// 防止越界
            endIdx = currentFood.itemCount;
    }

    item_y = (page == 0) ? 450 : 200;
    for (i = startIdx; i < endIdx; i++) {
        char total_str[100]; // 商品总价
        char quantity_str[100]; // 商品数量

        if(type == ORDER_SUPERMARKET)//超市订单
        {
```

```
                    int quantity = currentOrder.item[i].quantity; //
商品数量
                    float price = currentOrder.item[i].price; // 商品
价格

                    sprintf(total_str, "%.2f", price * quantity);
                    sprintf(quantity_str, "x%d", quantity);

                      PrintCC(200, item_y, currentOrder.item[i].name,
HEI, 24, 1, black); // 商品名
                }
                else if(type==ORDER_FOOD)//食堂订单
                {
                    int quantity = currentFood.item[i].quantity; // 商
品数量
                    float price = currentFood.item[i].price; // 商品价
格

                    sprintf(total_str, "%.2f", price * quantity);
                    sprintf(quantity_str, "x%d", quantity);

                       PrintCC(200, item_y, currentFood.item[i].name,
HEI, 24, 1, black); // 商品名
                }

                 PrintText(650, item_y, (unsigned char*)quantity_str,
HEI, 24, 1, black);// 商品数量
                PrintText(800, item_y, (unsigned char*)total_str, HEI,
24, 1, black);// 商品总价

                item_y += 50;
            }
```

```
            // 判断是否需要在此页显示总金额（当前页没有满）
            fullPageItemCount = (page == 0) ? 4 : 9;// 第一页显示 4 个商品，其余页显示 9 个商品

            if (type == ORDER_SUPERMARKET)
            {
                                if ((endIdx - startIdx) < fullPageItemCount||endIdx==currentOrder.itemCount) {// 当前页商品数量不满一页或最后一个商品刚好满页都要打印出总金额
                    //如果不是最后一个商品但是满页就不打印总金额
                    // 打印分隔线
                     PrintText(200, item_y - 30, "----------------------------------", HEI, 32, 1, black);

                    // 计算总金额
                    total_amount = 0.0;
                    for (i = 0; i < currentOrder.itemCount; i++) {
                         int quantity = currentOrder.item[i].quantity; // 商品数量
                          float price = currentOrder.item[i].price; // 商品价格
                        total_amount += price * quantity;
                    }

                    sprintf(total_str, "总金额:%.2f 元", total_amount);
                    PrintText(650, item_y + 10, total_str, HEI, 24, 1, black);

                }
            }
            else
            {
                                if ((endIdx - startIdx) <
```

```
fullPageItemCount||endIdx==currentFood.itemCount) {// 当前页商品数量不满一页或最后一个商品刚好满页都要打印出总金额
                    //如果不是最后一个商品但是满页就不打印总金额
                    // 打印分隔线
                     PrintText(200, item_y - 30, "----------------------------------------", HEI, 32, 1, black);

                    // 计算总金额
                    total_amount = 0.0;
                    for (i = 0; i < currentFood.itemCount; i++) {
                         int quantity = currentFood.item[i].quantity; // 商品数量
                        float price = currentFood.item[i].price; // 商品价格
                        total_amount += price * quantity;
                    }

                    sprintf(total_str, "总金额:%.2f 元", total_amount);
                    PrintText(750, item_y + 10, total_str, HEI, 24, 1, black);

                }
            }
        }
        if (route_state.picked[index] == 1)
            PrintCC(600, 715, "已取餐，待配送", HEI, 24, 1, Red);
        else
            PrintCC(600, 715, "待取餐", HEI, 24, 1, Red);

    }
/*************************************************************/
    #include "all_func.h"
```

```
    #define MAX_ACCEPTED_ORDERS 4
    #define ORDERS_PER_PAGE   4

    #define ORDER_SUPERMARKET 0
    #define ORDER_FOOD        1
    #define ORDER_DELIVER     2

    AcceptedOrder cur_orders[4]={0};

    // 接单处理：从对应展示列表中移除，并加入 cur_orders
    void  add_my_accept(OrderList  *OL,  FoodList  *FL,  DeliverList
*DL,int type, int local_index)
    {
        int i,distance_m;
        float item_price,distance_km;;
       //防止越界
        cur_orders[num_of_orders.cur_count].type = type;

        if (type == ORDER_SUPERMARKET)
        {
                    cur_orders[num_of_orders.cur_count].data.order   =
OL->elem[local_index];
                                                distance_m          =
dijkstra(&node[cur_orders[num_of_orders.cur_count].data.order.pick_u
p_location],
                &node[cur_orders[num_of_orders.cur_count].data.order.
destination],3); // 计算距离
           distance_km = distance_m / 1000.0; // 转换为公里
                                                item_price          =
cur_orders[num_of_orders.cur_count].data.order.total_amount;
            cur_orders[num_of_orders.cur_count].deliver_price=rider_d
eliver_price(distance_m, item_price);
           // 从超市订单列表移除该单
```

```
            // 先把所有后续元素往前移一位，再把订单表长减一
            for (i = local_index; i < OL->length - 1; i++)
                OL->elem[i] = OL->elem[i + 1];
            OL->length--;
            save_OL(OL);

        }
        else if (type == ORDER_FOOD)
        {
            // char debg[20];
            // sprintf(debg,"%d",local_index);
            // PrintText(50,0,debg,HEI,32,1,black);
                                                                        //
sprintf(debg,"%d",cur_orders[num_of_orders.cur_count].data.food.stat
ion);
            // PrintText(0,0,debg,HEI,32,1,black);
                    cur_orders[num_of_orders.cur_count].data.food   =
FL->elem[local_index];
                                                distance_m          =
dijkstra(&node[cur_orders[num_of_orders.cur_count].data.food.station
],
                &node[cur_orders[num_of_orders.cur_count].data.food.d
estination],3); // 计算距离
            distance_km = distance_m / 1000.0; // 转换为公里
                                                item_price          =
cur_orders[num_of_orders.cur_count].data.food.total_amount;
             cur_orders[num_of_orders.cur_count].deliver_price=rider_d
eliver_price(distance_m, item_price);
            // 从食品订单列表移除该单
            for (i = local_index; i < FL->length - 1; i++)
                FL->elem[i] = FL->elem[i + 1];
            FL->length--;
            save_FL(FL);
```

```
        }
        else if (type == ORDER_DELIVER)
        {
                  cur_orders[num_of_orders.cur_count].data.deliver  =
DL->elem[local_index];
                                                    distance_m          =
dijkstra(&node[cur_orders[num_of_orders.cur_count].data.deliver.stat
ion],
                &node[cur_orders[num_of_orders.cur_count].data.delive
r.index],3); // 计算距离
            distance_km = distance_m / 1000.0; // 转换为公里
            item_price = 2.0;
             cur_orders[num_of_orders.cur_count].deliver_price=rider_d
eliver_price(distance_m, item_price);
            // 从快递订单列表移除该单
            for (i = local_index; i < DL->length - 1; i++)
                DL->elem[i] = DL->elem[i + 1];
            DL->length--;
            save_DL(DL);
        }
        num_of_orders.cur_count++; //当前接单增加
        num_of_orders.total_cnt--; //展示订单减少
    }

    //从我的列表中移除
    void delete_my_order(int index)
    {
        int type;
        // OrderList OL = {0}; //创建超市，外卖，订单线性空表，便于读取
        // FoodList FL = {0};
        // DeliverList DL = {0};
        // ReadAllOrder(&OL); // 读取订单列表
        // ReadAllFood(&FL); // 读取食品列表
```

```
    // ReadAllDeliver(&DL); // 读取快递列表
    type = cur_orders[index].type;
    if (type == ORDER_SUPERMARKET)
    {
        save_order(cur_orders[index].data.order);
    }

    if (type == ORDER_FOOD)
    {
        save_food(cur_orders[index].data.food);
    }

    if (type == ORDER_DELIVER)
    {
        save_Deliver(cur_orders[index].data.deliver);
    }
    cut_current_order(index);
}

void my_accept_order(int user_pos)
{
    mouse_off_arrow(&mouse);
    draw_my_accept();
    mouse_on_arrow(mouse);
    while (1)
    {
        mouse_show_arrow(&mouse);
        if(mouse_press(122, 50, 242, 100)==1) //返回
        {
            mouse_off_arrow(&mouse);
            return;
            //my_information(users.pos);
        }
```

```
else if(mouse_press(342, 50, 462, 100)==1) //接单
{
    press3(1);//按钮高亮
    mouse_off_arrow(&mouse);
    accept_order(user_pos);//接单页面
    //return 后从这开始
    press3(3);
    press4(2);
    mouse_off_arrow(&mouse);
    bar1(0, 150, 1024, 768, white); // 清屏
    draw_my_accept();
    mouse_on_arrow(mouse);
}
else if(mouse_press(562, 50, 682, 100)==1) //路线
{
    press3(2); //按钮高亮
    mouse_off_arrow(&mouse);
    route(cur_orders,num_of_orders.cur_count,user_pos);
    //return 后从这开始
    press3(3);
    press4(2);
    mouse_on_arrow(mouse);
    bar1(0, 150, 1024, 768, white);
    draw_my_accept();
    mouse_on_arrow(mouse);
}
else if(mouse_press(782, 50, 902, 100)==1) //我的
{
    press3(3); //按钮高亮
    mouse_off_arrow(&mouse);
    bar1(0,150,200,768,deepblue);
    my_information(user_pos);
    //return 后从这开始
```

```
            press3(3);
            press4(2);
            mouse_on_arrow(mouse);
            bar1(0, 150, 1024, 768, white);
            draw_my_accept();
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(40, 276, 160, 326) == 1) //信息
        {
            press4(1);
            my_information(user_pos);
            //return 后从这开始
            press3(3);
            press4(2);
            mouse_on_arrow(mouse);
            bar1(0, 150, 1024, 768, white); //清屏
            draw_my_accept();
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(40, 602, 160, 652) == 1) //历史
        {
            press4(3);
            my_history_order(user_pos);
            //return 后从这开始
            press3(3);
            press4(2);
            mouse_on_arrow(mouse);
            bar1(0, 150, 1024, 768, white); // 清除路线界面残留
            draw_my_accept();
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(750, 195, 850, 245) == 1) //点击订单 1
详情
```

```
            {
                my_accept_detail(0,user_pos);
                //return 后从这开始
                press3(3);
                press4(2);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); //
                draw_my_accept();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(750, 315, 850, 365)==1) //点击订单 2 详
情
            {
                my_accept_detail(1,user_pos);
                //return 后从这开始
                press3(3);
                press4(2);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); //
                draw_my_accept();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(750, 435, 850, 485) == 1) //点击订单 3
详情
            {
                my_accept_detail(2,user_pos);
                //return 后从这开始
                press3(3);
                press4(2);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); //
                draw_my_accept();
                mouse_on_arrow(mouse);
```

```
            }
            else if(mouse_press(750, 555, 850, 605) == 1) //点击订单 4
详情
            {
                my_accept_detail(3,user_pos);
                //return 后从这开始
                press3(3);
                press4(2);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); //
                draw_my_accept();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(875, 195, 975, 245) == 1) //点击取消订
单 1
            {
                delete_my_order(0);
                //删除后更新我的当前列表
                press3(3);
                press4(2);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white);
                draw_my_accept();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(875, 315, 975, 365)==1) //点击取消订单
2
            {
                delete_my_order(1);
                //删除后更新我的当前列表
                press3(3);
                press4(2);
                mouse_on_arrow(mouse);
```

```
                bar1(0, 150, 1024, 768, white); //
                draw_my_accept();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(875, 435, 975, 485) == 1) //点击取消订
单 3
            {
                delete_my_order(2);
                //删除后更新我的当前列表
                mouse_on_arrow(mouse);
                press3(3);
                press4(2);
                bar1(0, 150, 1024, 768, white); //
                draw_my_accept();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(875, 555, 975, 605) == 1) //点击取消订
单 4
            {
                delete_my_order(3);
                //删除后更新我的当前列表
                press3(3);
                press4(2);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); //
                draw_my_accept();
                mouse_on_arrow(mouse);
            }
        }
    }

    void draw_my_accept()
    {
```

```
int y_offset = 170;
char pick_up[100], dest[100], distance_str[50], price_str[50];
int distance_m;
float dist_km, amount, fee;
int i;
//char debg[20];
bar1(200, 150, 1024, 768, white);
bar1(0, 150, 200, 768,deepblue);

Fill_Rounded_Rectangle(40, 276, 160, 326, 25,white);//填色
Fill_Rounded_Rectangle(40, 439, 160, 489, 25,deepblue);//填色
Fill_Rounded_Rectangle(40, 602, 160, 652, 25,white);//填色


Draw_Rounded_Rectangle(40, 276, 160, 326, 25, 1,deepblue);//
信息按钮
Draw_Rounded_Rectangle(40, 439, 160, 489, 25, 1,white);//当前
按钮
Draw_Rounded_Rectangle(40, 602, 160, 652, 25, 1,deepblue);//
历史按钮

PrintCC(75,291,"信息",HEI,24,1,deepblue);
PrintCC(75,454,"当前",HEI,24,1,white);
PrintCC(75,617,"历史",HEI,24,1,deepblue);

//sprintf(debg,"%d",num_of_orders.cur_count);
//PrintText(150, 100, debg, HEI, 24, 1, Red);

if (num_of_orders.cur_count == 0)
PrintCC(400,400,"当前无正在进行中订单",HEI,32,1,Red);
else
{
    for (i = 0; i < num_of_orders.cur_count; i++)
```

```
            {
                AcceptedOrder *ao = &cur_orders[i];
                // 画框
                 Draw_Rounded_Rectangle(220, y_offset, 1000, y_offset
+ 100, 30, 1, 0x6B4D);

                    Fill_Rounded_Rectangle(750,  y_offset+25,  850,
y_offset+75, 25, white);
                    Draw_Rounded_Rectangle(750,  y_offset+25,  850,
y_offset+75, 25, 1,deepblue);
                  PrintCC(750+25,  y_offset+35,  "详情",  HEI,  24,  1,
deepblue);

                // 取消按钮
                 Fill_Rounded_Rectangle(875,  y_offset  +  25,  975,
y_offset + 75, 25, white);
                 Draw_Rounded_Rectangle(875,  y_offset  +  25,  975,
y_offset + 75, 25, 1, deepblue);
                PrintCC(900,  y_offset  +  35,  "取消",  HEI,  24,  1,
deepblue);
                // 根据类型取数据
                switch (ao->type) {
                    case ORDER_SUPERMARKET: {
                        int pu   = ao->data.order.pick_up_location;
                        int dst  = ao->data.order.destination;
                        amount   = ao->data.order.total_amount;

                        sprintf(pick_up, "取货点：%s", node[pu].name);
                        sprintf(dest,    "目的地：%s", node[dst].name);
                          distance_m = dijkstra(&node[pu], &node[dst],
3);
                        break;
                    }
```

```
                case ORDER_FOOD: {
                    int pu   = ao->data.food.station;
                    int dst  = ao->data.food.destination;
                    amount   = ao->data.food.total_amount;

                    sprintf(pick_up, "取餐点：%s", node[pu].name);
                    sprintf(dest,    "目的地：%s", node[dst].name);
                     distance_m = dijkstra(&node[pu], &node[dst],
3);
                    break;
                }
                case ORDER_DELIVER: {
                    int pu   = ao->data.deliver.station+408;
                    int dst  = ao->data.deliver.index;
                    amount   = 2.0f;

                    // sprintf(debg,"deliver_station=%d",pu);
                    // PrintText(200, 50, debg, HEI, 24, 1, 0x0000);
                    sprintf(pick_up, "取货点：%s", node[pu].name);
                    sprintf(dest,    "目的地：%s", node[dst].name);
                     distance_m = dijkstra(&node[pu], &node[dst],
3);
                    break;
                }
                default:
                    continue;
            }

            // 计算并打印距离、费用
            dist_km = distance_m / 1000.0f;
            fee     = rider_deliver_price(distance_m, amount);

            PrintText(250,  y_offset + 10, pick_up, HEI, 24, 1, BLACK);
```

```
            PrintText(250,  y_offset + 60, dest,    HEI, 24, 1, BLACK);
            sprintf(distance_str, "距离：%.2fkm", dist_km);
             PrintText(500, y_offset + 10, distance_str, HEI, 24, 1,
BLACK);
            sprintf(price_str,    "配送费：%.1f 元", fee);
             PrintText(500, y_offset + 60, price_str,    HEI, 24, 1,
BLACK);


            y_offset += 120;
        }
        }

    }

    void cut_current_order(int index) //从列表中删除某个索引
    {
        int i;
        for ( i = index; i < num_of_orders.cur_count - 1; i++)
        {
            cur_orders[i] = cur_orders[i+1];//后面的订单将前面的覆盖
             route_state.picked[i]   = route_state.picked[i+1];// 同步
移动 picked[] / delivered[] 数组
            route_state.delivered[i]= route_state.delivered[i+1];
        }
        num_of_orders.cur_count--;
    }

/*****************************************************************/
    #include <all_func.h>

    #define MAX_ACCEPTED_ORDERS 4
    #define ORDERS_PER_PAGE   4
```

```
#define ORDER_SUPERMARKET 0
#define ORDER_FOOD        1
#define ORDER_DELIVER     2

AcceptedOrder hst_orders[10]={0};

void my_history_order(int user_pos)
{
    int page = 0;
    int type, local, global;
    mouse_off_arrow(&mouse);
    draw_my_history_order();
    mouse_on_arrow(mouse);

    while (1) {
        mouse_show_arrow(&mouse);
        if(mouse_press(122, 50, 242, 100)==1) //返回
        {
            mouse_off_arrow(&mouse);
            return;
            //my_information;
        }
        else if(mouse_press(342, 50, 462, 100)==1)
        {
            press3(1);//进入接单界面
            mouse_off_arrow(&mouse);
            accept_order(user_pos);//接单页面
            //return 后从这开始
            press3(3);
            press4(3);
            mouse_off_arrow(&mouse);
            bar1(0, 150, 1024, 768, white); // 清除接单界面残留
```

```
            draw_my_history_order();
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(562, 50, 682, 100)==1) //路线
        {
            press3(2); //按钮高亮
            mouse_off_arrow(&mouse);
            bar1(0, 150, 1024, 768, white); // 清除接单界面残留
             route(cur_orders,num_of_orders.cur_count,user_pos);//
骑手路线规划
            //return 后从这开始
            press3(3);
            press4(3);
            mouse_on_arrow(mouse);
            bar1(0, 150, 1024, 768, white); // 清除路线界面残留
            draw_my_history_order(); // 重新绘制订单列表
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(782, 50, 902, 100)==1) //我的
        {
            press3(3); //按钮高亮
            mouse_off_arrow(&mouse);
            my_information(user_pos);
            //return 后从这开始
            press3(3);
            press4(3);
            mouse_on_arrow(mouse);
            bar1(0, 150, 1024, 768, white); // 清除路线界面残留
            draw_my_history_order(); // 重新绘制订单列表
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(40, 276, 160, 326) == 1) //信息
        {
```

```
    press4(1);
    my_information(user_pos);
    //return 后从这开始
    press3(3);
    press4(3);
    mouse_on_arrow(mouse);
    bar1(0, 150, 1024, 768, white); //清屏
    draw_my_accept();
    mouse_on_arrow(mouse);
}
else if(mouse_press(40, 439, 160, 489) == 1) //当前
{
    press4(2);
    my_accept_order(user_pos);
    //return 后从这开始
    press3(3);
    press4(3);
    mouse_on_arrow(mouse);
    bar1(0, 150, 1024, 768, white); // 清除路线界面残留
    draw_my_history_order();
    mouse_on_arrow(mouse);
}
else if(mouse_press(40, 602, 160, 652) == 1) //历史
{
    press4(3);
    my_history_order(user_pos);
    //return 后从这开始
    press3(3);
    press4(3);
    mouse_on_arrow(mouse);
    bar1(0, 150, 1024, 768, white); // 清除路线界面残留
    draw_my_history_order();
    mouse_on_arrow(mouse);
```

```
            }
            else if(mouse_press(750, 195, 850, 245) == 1) //点击订单 1
详情
            {
                my_history_detail(0,user_pos);
                //return 后从这开始
                press3(3);
                press4(3);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); //
                draw_my_history_order();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(750, 315, 850, 365)==1) //点击订单 2 详
情
            {
                my_history_detail(1,user_pos);
                //return 后从这开始
                press3(3);
                press4(3);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); //
                draw_my_history_order();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(750, 435, 850, 485) == 1) //点击订单 3
详情
            {
                my_history_detail(2,user_pos);
                //return 后从这开始
                press3(3);
                press4(3);
                mouse_on_arrow(mouse);
```

```
                bar1(0, 150, 1024, 768, white); //
                draw_my_history_order();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(750, 555, 850, 605) == 1) //点击订单 4
详情
            {
                my_history_detail(3,user_pos);
                //return 后从这开始
                press3(3);
                press4(3);
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white); //
                draw_my_history_order();
                mouse_on_arrow(mouse);
            }
        }
    }

    void move_to_history(AcceptedOrder cur_orders[], int order_idx)
    {
        int j;
        if (num_of_orders.hst_count < 10 && num_of_orders.cur_count >=
0 && order_idx < num_of_orders.cur_count) \
        {
            // 复制到历史列表末尾
                            hst_orders[num_of_orders.hst_count++]    =
cur_orders[order_idx];
            // 从 cur_orders 中移除：后续元素前移
            for ( j = order_idx; j < num_of_orders.cur_count - 1; j++)
                cur_orders[j] = cur_orders[j + 1];
            num_of_orders.cur_count--;
        }
```

```
    }

    void draw_my_history_order()
    {
        int y_offset = 170;
        char pick_up[100], dest[100], distance_str[50], price_str[50];
        int distance_m;
        float dist_km, amount, fee;
        int i;
        //char debg[20];
        bar1(200, 150, 1024, 768, white);
        //sprintf(debg,"%d",num_of_orders.hst_count);
        //PrintText(0,  0, debg, HEI, 24, 1, BLACK);
        if (num_of_orders.hst_count == 0)
        PrintCC(450,400,"当前无历史订单",HEI,32,1,Red);
        else
        {
            for (i = 0; i < num_of_orders.hst_count; i++)
            {
                AcceptedOrder *ho = &hst_orders[i];
                // 画框
                 Draw_Rounded_Rectangle(220, y_offset, 1000, y_offset
+ 100, 30, 1, 0x6B4D);

                    Fill_Rounded_Rectangle(750,  y_offset+25,  850,
y_offset+75, 25, white);
                    Draw_Rounded_Rectangle(750,  y_offset+25,  850,
y_offset+75, 25, 1,deepblue);
                  PrintCC(750+25,  y_offset+35,  "详情",  HEI,  24,  1,
deepblue);

                // // 取消按钮
                  // Fill_Rounded_Rectangle(875, y_offset + 25, 975,
```

```
y_offset + 75, 25, white);
                  // Draw_Rounded_Rectangle(875, y_offset + 25, 975,
y_offset + 75, 25, 1, deepblue);
                  // PrintCC(900, y_offset + 35, "取消", HEI, 24, 1,
deepblue);
                // 根据类型取数据
                switch (ho->type) {
                    case ORDER_SUPERMARKET: {
                        int pu   = ho->data.order.pick_up_location;
                        int dst  = ho->data.order.destination;
                        amount   = ho->data.order.total_amount;

                        sprintf(pick_up, "取货点：%s", node[pu].name);
                        sprintf(dest,    "目的地：%s", node[dst].name);
                         distance_m = dijkstra(&node[pu], &node[dst],
3);
                        break;
                    }
                    case ORDER_FOOD: {
                        int pu   = ho->data.food.station;
                        int dst  = ho->data.food.destination;
                        amount   = ho->data.food.total_amount;

                        sprintf(pick_up, "取餐点：%s", node[pu].name);
                        sprintf(dest,    "目的地：%s", node[dst].name);
                         distance_m = dijkstra(&node[pu], &node[dst],
3);
                        break;
                    }
                    case ORDER_DELIVER: {
                        int pu   = ho->data.deliver.station+408;
                        int dst  = ho->data.deliver.index;
                        amount   = 2.0f;
```

```
                    sprintf(pick_up, "取货点：%s", node[pu].name);
                    sprintf(dest,    "目的地：%s", node[dst].name);
                     distance_m = dijkstra(&node[pu], &node[dst],
3);
                    break;
                }
                default:
                    continue;
            }

            // 计算并打印距离、费用
            dist_km = distance_m / 1000.0f;
            fee     = rider_deliver_price(distance_m, amount);

             PrintText(250,  y_offset + 10, pick_up, HEI, 24, 1,
BLACK);
             PrintText(250,  y_offset + 60, dest,    HEI, 24, 1,
BLACK);
            sprintf(distance_str, "距离：%.2fkm", dist_km);
             PrintText(500, y_offset + 10, distance_str, HEI, 24,
1, BLACK);
            sprintf(price_str,    "配送费：%.1f 元", fee);
             PrintText(500, y_offset + 60, price_str,    HEI, 24,
1, BLACK);
            y_offset += 120;
        }
    }

}

void draw_my_history_detail_body(int type, int index)
{
    char buf[128];
```

```
        // 通用：下单时间、手机号
        char time_str[64];
        char phone_str[64];
        char show_distance[30];
        char show_deliver_price[30];
        int distance_m;
        float distance_km;
        float item_price;
        float deliver_price;

        if (type == ORDER_SUPERMARKET)
        {
                        sprintf(time_str,   "  下  单  时  间  :  %s",
hst_orders[index].data.order.order_time);
                         sprintf(phone_str,   "  手  机  号  :  %s",
hst_orders[index].data.order.user_phone);
            // 取货点
                           sprintf(buf,    "  取  货  点  :  %s",
node[hst_orders[index].data.order.pick_up_location].name);
            PrintText(200, 150 + 150, buf, HEI, 24, 1, black);
            // 用户地址
                         sprintf(buf,    "  用  户  地  址  :  %s",
node[hst_orders[index].data.order.destination].name);
            PrintText(500, 150 + 150, buf, HEI, 24, 1, black);
            // 距离
                                                distance_m          =
dijkstra(&node[hst_orders[index].data.order.pick_up_location],
&node[hst_orders[index].data.order.destination],3); // 计算距离
            distance_km = distance_m / 1000.0; // 转换为公里
            sprintf(show_distance, "距离：%.2fkm", distance_km);
            PrintText(200, 200 + 150, show_distance, HEI, 24, 1, black);

              item_price  =  hst_orders[index].data.order.total_amount;
```

```
// 获取商品价格
                    deliver_price   =   rider_deliver_price(distance_m,
item_price); // 计算配送费用
             hst_orders[index].deliver_price=deliver_price; //将此单配
送费存进结构体中
                 sprintf(show_deliver_price,  "配送费：%.1f 元",
deliver_price);
             PrintText(500, 200+150, show_deliver_price, HEI, 24, 1,
black);


        }
        else if (type == ORDER_FOOD)
        {

                       sprintf(time_str,   "下单时间：%s",
hst_orders[index].data.food.order_time);
                        sprintf(phone_str,   "手机号：%s",
hst_orders[index].data.food.user_phone);
            // 取餐点
                           sprintf(buf,    "取货点：%s",
node[hst_orders[index].data.food.station].name);
            PrintText(200, 150 + 150, buf, HEI, 24, 1, black);
            // 用户地址
                        sprintf(buf,    "用户地址：%s",
node[hst_orders[index].data.food.destination].name);
            PrintText(500, 150 + 150, buf, HEI, 24, 1, black);
            //距离
                                                distance_m          =
dijkstra(&node[hst_orders[index].data.food.station],
&node[hst_orders[index].data.food.destination],3); // 计算距离
            distance_km = distance_m / 1000.0; // 转换为公里
            sprintf(show_distance, "距离：%.2fkm", distance_km);
            PrintText(200, 200 + 150, show_distance, HEI, 24, 1, black);
```

```
            item_price = hst_orders[index].data.order.total_amount;
// 获取商品价格
                deliver_price = rider_deliver_price(distance_m,
item_price); // 计算配送费用
          hst_orders[index].deliver_price=deliver_price;
               sprintf(show_deliver_price, "配送费：%.1f 元",
deliver_price);
           PrintText(500, 200+150, show_deliver_price, HEI, 24, 1,
black);
       }
       else if (type == ORDER_DELIVER)
       {
                   sprintf(time_str, "下单时间：%s",
hst_orders[index].data.deliver.time);
                    sprintf(phone_str, "手机号：%s",
hst_orders[index].data.deliver.number);
          // 取货点
                      sprintf(buf, "取货点：%s",
node[hst_orders[index].data.deliver.station].name);
          PrintText(200, 150 + 150, buf, HEI, 24, 1, black);
          // 用户地址
                    sprintf(buf, "用户地址：%s",
node[hst_orders[index].data.deliver.destination].name);
          PrintText(500, 150 + 150, buf, HEI, 24, 1, black);
          //距离
                                              distance_m =
dijkstra(&node[hst_orders[index].data.deliver.station],
&node[hst_orders[index].data.deliver.destination],3); // 计算距离
          distance_km = distance_m / 1000.0; // 转换为公里
          sprintf(show_distance, "距离：%.2fkm", distance_km);
            PrintText(200, 200 + 150, show_distance, HEI, 24, 1,
0x0000);
```

```
            item_price = 2.0; // 获取商品价格
                   deliver_price   =   rider_deliver_price(distance_m,
item_price); // 计算配送费用
            hst_orders[index].deliver_price=deliver_price;
                 sprintf(show_deliver_price,  "配送费：%.1f 元",
deliver_price);
             PrintText(500, 200+150, show_deliver_price, HEI, 24, 1,
0x0000);
        }
        // 打印通用字段（相对偏移）
        PrintText(200, 50  + 150, time_str, HEI, 24, 1, black);
        PrintText(200, 100 + 150, phone_str, HEI, 24, 1, black);
    }

    void my_history_detail(int index , int user_pos)
    {
        OrderList OL = {0};
        FoodList FL = {0};
        int page = 0;
        int totalPage = 1;

        int type;
        type=hst_orders[index].type;

        ReadAllOrder(&OL); // 读取订单列表
        ReadAllFood(&FL); // 读取食品列表

        // 计算总页数//第一页 4 个，第二页后面 9 个
        if (type == ORDER_SUPERMARKET) {
            totalPage = (OL.elem[index].itemCount - 4 + 8) / 9 + 1;
        } else if (type == ORDER_FOOD) {
            totalPage = (FL.elem[index].itemCount - 4 + 8) / 9 + 1;
```

```
        }
        // ORDER_DELIVER 保持 totalPage=0

        DestroyOList(&OL); // 释放快递列表内存
        DestroyFList(&FL); // 释放食品列表内存

        draw_my_history_detail(type, index,page);

        mouse_on_arrow(mouse);
        while (1)
        {
            mouse_show_arrow(&mouse);
            // 返回按钮
            if(mouse_press(122, 50, 242, 100)==1) //返回
            {
                return;
                //my_accept_order(users.pos);
            }
            else if(mouse_press(562, 50, 682, 100)==1) //路线
            {
                press3(2); //按钮高亮
                mouse_off_arrow(&mouse);
                bar1(0, 150, 1024, 768, white);
                                                    route(cur_orders,
num_of_orders.cur_count,user_pos);//骑手路线规划
                //return 后从这开始
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white);
                draw_my_order_detail(type, index,page);
            }
            else if(mouse_press(782, 50, 902, 100)==1) //我的
            {
                press3(3); //按钮高亮
```

```
                mouse_off_arrow(&mouse);
                bar1(0, 150, 1024, 768, white);
                my_information(user_pos);
                //return 后从这开始
                mouse_on_arrow(mouse);
                bar1(0, 150, 1024, 768, white);
                draw_my_order_detail(type, index,page);
            }
            else if (mouse_press(220, 700, 340, 750) == 1)
            {
                if (page > 0) {
                    page--;
                    draw_my_order_detail(type, index,page);
                } else {
                    // 提示：已是第一页
                         PrintCC(630, 715, "已是第一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(630, 715, 780, 765, white);
                }
            }
            else if (mouse_press(420, 700, 540, 750) == 1)
            {
                if (page < totalPage - 1) {
                    page++;
                    draw_my_order_detail(type, index,page);
                } else {
                    // 提示：已是最后一页
                        PrintCC(630, 715, "已是最后一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(630, 715, 780, 765, white);
                }
```

```
        }

    }

}

void draw_my_history_detail(int type,int index,int page)
{
    int i;
    Order currentOrder ;
    FoodOrder currentFood;
    Deliver currentDeliver;

    char current_time[100]; // 获取当前时间
    char time_str[100]; // 打印下单时间
    char user_name[100]; // 打印用户名
    char user_phone[100]; // 打印用户手机号
    char order_number; // 打印订单号
    char address[100]; // 打印用户地址
    int startIdx = 0;// 起始商品索引
    int itemsPerPage = 0;// 每页商品数量
    int endIdx = 0;// 结束商品索引
    int item_y = 0;// 商品框的 y 坐标

    float total_amount = 0.0; // 总金额
    char total_str[100]; // 总金额字符串
    int fullPageItemCount = 0; // 满页商品数量
    // 清屏
    bar1(0, 150, 1024, 768, white);

    currentOrder = hst_orders[index].data.order; // 当前订单
    currentFood = hst_orders[index].data.food; // 当前订单
    currentDeliver = hst_orders[index].data.deliver; // 当前快递
```

```
        // 绘制主体部分


        // 分页按钮（超市和外卖类型）
        if (type != ORDER_DELIVER) {
                Draw_Rounded_Rectangle(220,  700,  340,  750,  25,  1,
deepblue);
            PrintCC(245, 715, "上一页", HEI, 24, 1, deepblue);
                Draw_Rounded_Rectangle(420,  700,  540,  750,  25,  1,
deepblue);
            PrintCC(445, 715, "下一页", HEI, 24, 1, deepblue);
        }

        // 第一页绘制头部
        if(page==0&&type!=ORDER_DELIVER)
        {
            draw_my_order_detail_body(type, index);

            // 表头
            PrintCC(200, 400, "商品详情", HEI, 24, 1, black);
            PrintCC(650, 400, "数量", HEI, 24, 1, black);
            PrintCC(800, 400, "金额", HEI, 24, 1, black);
            PrintText(200, 420, "------------------------------", HEI,
32, 1, black);

            startIdx = 0;
            itemsPerPage = 4;//第一页 4 个，第二页后面 9 个
        }
        else if(page==0&&type==ORDER_DELIVER)
        {
            draw_my_order_detail_body(type, index);
        }
```

```
else // 其他页
{
    startIdx = 4 + (page - 1) * 9;
    itemsPerPage = 9;
}

endIdx = startIdx + itemsPerPage;

if (type == ORDER_DELIVER) {
    // 只展示取件码
    char code_buf[64];
    sprintf(code_buf, "取件码：%s", currentDeliver.code);
    PrintText(200, 400, code_buf, HEI, 32, 1, black);
} else {


    if (type == ORDER_SUPERMARKET)//超市订单
    {
        if (endIdx > currentOrder.itemCount)// 防止越界
            endIdx = currentOrder.itemCount;
    }
    else if(type==ORDER_FOOD)//食堂订单
    {
        if (endIdx > currentFood.itemCount)// 防止越界
            endIdx = currentFood.itemCount;
    }

    item_y = (page == 0) ? 450 : 200;
    for (i = startIdx; i < endIdx; i++) {
        char total_str[100]; // 商品总价
        char quantity_str[100]; // 商品数量

        if(type == ORDER_SUPERMARKET)//超市订单
```

```
            {
                 int quantity = currentOrder.item[i].quantity; //
商品数量
                 float price = currentOrder.item[i].price; // 商品
价格

                sprintf(total_str, "%.2f", price * quantity);
                sprintf(quantity_str, "x%d", quantity);

                  PrintCC(200, item_y, currentOrder.item[i].name,
HEI, 24, 1, black); // 商品名
            }
            else if(type==ORDER_FOOD)//食堂订单
            {
                int quantity = currentFood.item[i].quantity; // 商
品数量
                float price = currentFood.item[i].price; // 商品价
格

                sprintf(total_str, "%.2f", price * quantity);
                sprintf(quantity_str, "x%d", quantity);

                   PrintCC(200, item_y, currentFood.item[i].name,
HEI, 24, 1, black); // 商品名
            }

             PrintText(650, item_y, (unsigned char*)quantity_str,
HEI, 24, 1, black);// 商品数量
            PrintText(800, item_y, (unsigned char*)total_str, HEI,
24, 1, black);// 商品总价

            item_y += 50;
        }
```

```
            // 判断是否需要在此页显示总金额（当前页没有满）
            fullPageItemCount = (page == 0) ? 4 : 9;// 第一页显示 4 个商品，其余页显示 9 个商品

            if (type == ORDER_SUPERMARKET)
            {
                                if ((endIdx - startIdx) < fullPageItemCount||endIdx==currentOrder.itemCount) {// 当前页商品数量不满一页或最后一个商品刚好满页都要打印出总金额
                    //如果不是最后一个商品但是满页就不打印总金额
                    // 打印分隔线
                     PrintText(200, item_y - 30, "----------------------------------------", HEI, 32, 1, black);

                    // 计算总金额
                    total_amount = 0.0;
                    for (i = 0; i < currentOrder.itemCount; i++) {
                         int quantity = currentOrder.item[i].quantity; // 商品数量
                          float price = currentOrder.item[i].price; // 商品价格
                        total_amount += price * quantity;
                    }

                    sprintf(total_str, "总金额:%.2f 元", total_amount);
                    PrintText(650, item_y + 10, total_str, HEI, 24, 1, black);

                }
            }
            else
            {
```

```
                                    if    ((endIdx    -    startIdx)    <
fullPageItemCount||endIdx==currentFood.itemCount) {// 当前页商品数量不
满一页或最后一个商品刚好满页都要打印出总金额
                    //如果不是最后一个商品但是满页就不打印总金额
                    // 打印分隔线
                     PrintText(200, item_y - 30, "----------------------
-----------", HEI, 32, 1, black);

                    // 计算总金额
                    total_amount = 0.0;
                    for (i = 0; i < currentFood.itemCount; i++) {
                         int quantity = currentFood.item[i].quantity;
// 商品数量
                        float price = currentFood.item[i].price; // 商
品价格
                        total_amount += price * quantity;
                    }

                    sprintf(total_str, "总金额:%.2f 元", total_amount);
                    PrintText(750, item_y + 10, total_str, HEI, 24, 1,
black);

                }
            }
        }
    }

/*****************************************************************/
    #include <all_func.h>

    void my_information(int user_pos)
    {
        UserList UL = {0};
```

```
USER currentUser;
mouse_off_arrow(&mouse);
ReadAllUser(&UL); // 读取用户列表
currentUser=UL.elem[user_pos];// 获取当前用户信息
DestroyUList(&UL); // 释放用户列表空间
draw_my_information(currentUser);
mouse_on_arrow(mouse);
while(1)
{
    mouse_show_arrow(&mouse);
    if(mouse_press(122, 50, 242, 100)==1)
    {
        mouse_off_arrow(&mouse);
        return;
        //rider(int pos)
    }
    else if(mouse_press(342, 50, 462, 100)==1) //接单
    {
        press3(1);//进入接单界面
        accept_order(user_pos);//接单页面
        //return 后从这开始
        press3(3);
        press4(1);
        mouse_off_arrow(&mouse);
        bar1(0, 150, 1024, 768, white); // 清除接单界面残留
        draw_my_information(currentUser);
        mouse_on_arrow(mouse);
    }
    else if(mouse_press(562, 50, 682, 100)==1)
    {
        mouse_off_arrow(&mouse);
        press3(2);//进入路线规划界面
         route(cur_orders,num_of_orders.cur_count,user_pos);//
```

进入路线规划界面

```
            //return 后从这开始
            press3(3);
            press4(1);
            mouse_off_arrow(&mouse);
            bar1(0, 150, 1024, 768, white); // 清除接单界面残留
            draw_my_information(currentUser);
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(782, 50, 902, 100)==1) //我的
        {
            mouse_off_arrow(&mouse);
            press3(3); //按钮高亮
            my_accept_order(user_pos);
            //return 后从这开始
            press3(3);
            press4(1);
            mouse_on_arrow(mouse);
            bar1(0, 150, 1024, 768, white); // 清除路线界面残留
            draw_my_information(currentUser);
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(40, 439, 160, 489) == 1)//当前
        {
            mouse_off_arrow(&mouse);
            press4(2);
            my_accept_order(user_pos);
            //return 后从这开始
            press3(3);
            press4(1);
            mouse_on_arrow(mouse);
            bar1(0, 150, 1024, 768, white); // 清除路线界面残留
            draw_my_information(currentUser);
```

```
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(40, 602, 160, 652) == 1)//历史
        {
            mouse_off_arrow(&mouse);
            press4(3);
            my_history_order(user_pos);
            //return 后从这开始
            press3(3);
            press4(1);
            mouse_on_arrow(mouse);
            bar1(0, 150, 1024, 768, white); // 清除路线界面残留
            draw_my_information(currentUser);
            mouse_on_arrow(mouse);
        }
    }
}

void draw_my_information(USER currentUser)
{
    char show_name[50];
    char show_num[50];
    char show_account[50];
    bar1(0, 150, 1024, 768,white);
    bar1(0, 150, 200, 1024, deepblue);
    sprintf(show_name,"用户：%s",currentUser.name);
    PrintText(350,200,show_name,HEI,32,1,black);
    sprintf(show_num,"手机号：%s",currentUser.number);
    PrintText(350,350,show_num,HEI,32,1,black);

    if(currentUser.ocp==1 && currentUser.type==3)
    PrintText(350,500,"骑手类型：全职骑手",HEI,32,1,black);
    else if(currentUser.ocp == 2 && currentUser.type == 3)
```

```
        PrintText(350,500,"骑手类型：兼职骑手",HEI,32,1,black);

        sprintf(show_account,"余额：%.2f",currentUser.account);
        PrintText(350,650,show_account,HEI,32,1,black);

        Fill_Rounded_Rectangle(40, 276, 160, 326, 25,deepblue);//填色
        Fill_Rounded_Rectangle(40, 439, 160, 489, 25,white);//填色
        Fill_Rounded_Rectangle(40, 602, 160, 652, 25,white);//填色


        Draw_Rounded_Rectangle(40, 276, 160, 326, 25, 1,white);//信息
按钮
        Draw_Rounded_Rectangle(40, 439, 160, 489, 25, 1,deepblue);//
当前按钮
        Draw_Rounded_Rectangle(40, 602, 160, 652, 25, 1,deepblue);//
历史按钮

        PrintCC(75,291,"信息",HEI,24,1,white);
        PrintCC(75,454,"当前",HEI,24,1,deepblue);
        PrintCC(75,617,"历史",HEI,24,1,deepblue);
    }

    void press4(int x)
    {
        mouse_off_arrow(&mouse);
        switch (x)
        {
            case 1:
            {
                    Fill_Rounded_Rectangle(40, 276, 160, 326, 25,
deepblue);
                Draw_Rounded_Rectangle(40, 276, 160, 326, 25, 1,white);
                PrintCC(75,291, "信息", HEI, 24, 1, white);
```

```
                Fill_Rounded_Rectangle(40, 439, 160, 489, 25, white);
                     Draw_Rounded_Rectangle(40,  439,  160,  489,  25,
1,deepblue);
                PrintCC(75,454, "当前", HEI, 24, 1, deepblue);
                Fill_Rounded_Rectangle(40, 602, 160, 652, 25, white);
                     Draw_Rounded_Rectangle(40,  602,  160,  652,  25,
1,deepblue);
                PrintCC(75,617, "历史", HEI, 24, 1, deepblue);
                break;
            }
            case 2:
            {
                Fill_Rounded_Rectangle(40, 276, 160, 326, 25, white);
                     Draw_Rounded_Rectangle(40,  276,  160,  326,  25,
1,deepblue);
                PrintCC(75, 291, "信息", HEI, 24, 1, deepblue);
                     Fill_Rounded_Rectangle(40,  439,  160,  489,  25,
deepblue);
                Draw_Rounded_Rectangle(40, 439, 160, 489, 25, 1,white);
                PrintCC(75, 454, "当前", HEI, 24, 1, white);
                Fill_Rounded_Rectangle(40, 602, 160, 652, 25, white);
                     Draw_Rounded_Rectangle(40,  602,  160,  652,  25,
1,deepblue);
                PrintCC(75, 617, "历史", HEI, 24, 1, deepblue);
                break;
            }
            case 3:
            {
                Fill_Rounded_Rectangle(40, 276, 160, 326, 25, white);
                     Draw_Rounded_Rectangle(40,  276,  160,  326,  25,
1,deepblue);
                PrintCC(75,291, "信息", HEI, 24, 1, deepblue);
                Fill_Rounded_Rectangle(40, 439, 160, 489, 25, white);
```

```
                    Draw_Rounded_Rectangle(40,  439,  160,  489,  25,
1,deepblue);
                PrintCC(75,454, "当前", HEI, 24, 1, deepblue);
                    Fill_Rounded_Rectangle(40,  602,  160,  652,  25,
deepblue);
                Draw_Rounded_Rectangle(40, 602, 160, 652, 25, 1,white);
                PrintCC(75, 617, "历史", HEI ,24 ,1 ,white);
                break;
            }
            default:
                break;
        }
        mouse_on_arrow(mouse);
    }
/*****************************************************************/
    #include "all_func.h"

    void rider(int user_pos){

        draw_rider();

        mouse_on_arrow(mouse);


        while(1){
            mouse_show_arrow(&mouse);

            if(mouse_press(122, 50, 242, 100)==1)
            {
                return ;
                //welcome()首页
            }
            else if(mouse_press(342, 50, 462, 100)==1)
```

```
        {
            press3(1);//进入接单界面
            accept_order();//接单页面

            //return 后从这开始
            mouse_off_arrow(&mouse);
            bar1(0, 150, 1024, 768, white); // 清除接单界面残留
            draw_rider();
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(562, 50, 682, 100)==1)
        {
            press3(2);//进入路线规划界面
            route(acp_orders,delivers.acp_count);//进入路线规划界面
            //return 后从这开始
            mouse_off_arrow(&mouse);
            bar1(0, 150, 1024, 768, white); // 清除接单界面残留
            draw_rider();
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(782, 50, 902, 100)==1) //我的
        {
            press3(3); //按钮高亮
            my_accept_order();
            //return 后从这开始
            mouse_on_arrow(mouse);
            bar1(0, 150, 1024, 768, white); // 清除路线界面残留
            draw_rider();
            mouse_on_arrow(mouse);
        }
    }
}
```

```
void draw_rider(){
    bar1(0, 0, 1024, 768, white);
    bar1(0, 0, 1024, 150, deepblue);


    Fill_Rounded_Rectangle(122, 50, 242, 100, 25,white);//填色
    Fill_Rounded_Rectangle(342, 50, 462, 100, 25,white);//填色
    Fill_Rounded_Rectangle(562, 50, 682, 100, 25,white);//填色
    Fill_Rounded_Rectangle(782, 50, 902, 100, 25,white);//填色

    Draw_Rounded_Rectangle(122, 50, 242, 100, 25, 1,deepblue);//
返回
    Draw_Rounded_Rectangle(342, 50, 462, 100, 25, 1,deepblue);//
接单
    Draw_Rounded_Rectangle(562, 50, 682, 100, 25, 1,deepblue);//
路线
    Draw_Rounded_Rectangle(782, 50, 902, 100, 25, 1,deepblue);//
账户

    PrintCC(122+35, 65, "返回", HEI, 24, 1, deepblue);
    PrintCC(342+35, 65, "接单", HEI, 24, 1, deepblue);
    PrintCC(562+35, 65, "路线", HEI, 24, 1, deepblue);
    PrintCC(782+35, 65, "我的", HEI, 24, 1, deepblue);

    PrintCC(10, 10, "当前账号类型为：骑手", HEI, 24, 1, white);

}

void press3(int x){
    mouse_off_arrow(&mouse);
    switch (x)
    {
        case 1:
```

```
            {

                    Fill_Rounded_Rectangle(342,  50,  462,  100,  25,
deepblue);
                    Draw_Rounded_Rectangle(342,  50,  462,  100,  25,
1,deepblue);
                PrintCC(342+35, 65, "接单", HEI, 24, 1, white);
                Fill_Rounded_Rectangle(562, 50, 682, 100, 25, white);
                    Draw_Rounded_Rectangle(562,  50,  682,  100,  25,
1,deepblue);
                PrintCC(562+35, 65, "路线", HEI, 24, 1, deepblue);
                Fill_Rounded_Rectangle(782, 50, 902, 100, 25, white);
                    Draw_Rounded_Rectangle(782,  50,  902,  100,  25,
1,deepblue);
                PrintCC(782+35, 65, "我的", HEI, 24, 1, deepblue);
                break;
            }
            case 2:
            {
                Fill_Rounded_Rectangle(342, 50, 462, 100, 25, white);
                    Draw_Rounded_Rectangle(342,  50,  462,  100,  25,
1,deepblue);
                PrintCC(342+35, 65, "接单", HEI, 24, 1, deepblue);
                    Fill_Rounded_Rectangle(562,  50,  682,  100,  25,
deepblue);
                    Draw_Rounded_Rectangle(562,  50,  682,  100,  25,
1,deepblue);
                PrintCC(562+35, 65, "路线", HEI, 24, 1, white);
                Fill_Rounded_Rectangle(782, 50, 902, 100, 25, white);
                    Draw_Rounded_Rectangle(782,  50,  902,  100,  25,
1,deepblue);
                PrintCC(782+35, 65, "我的", HEI, 24, 1, deepblue);
                break;
```

```
            }
            case 3:
            {
                Fill_Rounded_Rectangle(342, 50, 462, 100, 25, white);
                     Draw_Rounded_Rectangle(342,  50,  462,  100,  25,
1,deepblue);
                PrintCC(342+35, 65, "接单", HEI, 24, 1, deepblue);
                Fill_Rounded_Rectangle(562, 50, 682, 100, 25, white);
                     Draw_Rounded_Rectangle(562,  50,  682,  100,  25,
1,deepblue);
                PrintCC(562+35, 65, "路线", HEI, 24, 1, deepblue);
                     Fill_Rounded_Rectangle(782,  50,  902,  100,  25,
deepblue);
                     Draw_Rounded_Rectangle(782,  50,  902,  100,  25,
1,deepblue);
                PrintCC(782+35, 65,"我的", HEI ,24 ,1 ,white);
                break;
            }
            }
        mouse_on_arrow(mouse);
    }
/*****************************************************************/
    #include <all_func.h>

    #define INF 20000
    #define MAX_NODES 417

    Node node [MAX_NODES] = {
        {0,0,{0,0,0,0,0,0}, {0,0,0,0,0,0}, 0,"0"},
         {854, 143, {131, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑食堂"},
        {971, 165, {406, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东园食堂"},
```

{857, 239, {366, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "学一食堂"},
{857, 228, {365, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "学二食堂"},
{880, 383, {153, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "东教工食堂"},
{381, 130, {384, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "喻园食堂"},
{451, 177, {246, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "集贤楼食堂"},
{521, 278, {197, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "东一食堂"},
{534, 284, {195, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "紫荆园餐厅"},
{549, 284, {194, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "东三食堂"},
{550, 269, {202, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "东四食堂"},
{125, 222, {385, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "西一食堂"},
{151, 214, {385, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "西二食堂"},
{120, 204, {386, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "西三食堂"},
{188, 152, {387, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "百景园餐厅"},
{380, 112, {254, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "集锦园餐厅"},
{114, 281, {329, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "百惠园餐厅"},
{858, 143, {130, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑超市"},
{713, 139, {154, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,

```
20000, 20000}, 1, "喻园超市"},
        {178, 189, {388, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西区超市"},
        {53, 283, {334, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 10 栋"},
        {81, 284, {332, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 6 栋"},
        {103, 286, {330, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 2 栋"},
        {39, 292, {335, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 14 栋"},
        {54, 296, {334, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 9 栋"},
        {81, 296, {332, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 5 栋"},
        {33, 301, {337, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 13 栋"},//
        {27, 308, {338, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 15 栋"},
        {37, 319, {339, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 12 栋"},
        {58, 316, {347, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 8 栋"},
        {81, 311, {355, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 4 栋"},
        {39, 331, {344, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 11 栋"},
        {59, 327, {348, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 7 栋"},
        {85, 322, {355, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 3 栋"},
        {106, 317, {382, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 1 栋"},//
```

```
        {115, 303, {353, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 16 栋"},
        {118, 320, {356, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 17 栋"},
        {114, 327, {357, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "紫菘 18 栋"},
        {164, 195, {317, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西一宿舍"},
        {160, 182, {317, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西二宿舍"},
        {142, 201, {316, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西三宿舍"},
        {135, 187, {316, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西四宿舍"},
        {211, 232, {283, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西五宿舍"},
        {158, 227, {390, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西六宿舍"},
        {249, 204, {408, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西七宿舍"},
        {219, 189, {296, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西八宿舍"},
        {211, 176, {301, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西九宿舍"},
        {195, 188, {303, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西十宿舍"},
        {191, 175, {391, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西十一宿舍"},
        {200, 201, {293, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西十二宿舍"},
        {202, 210, {288, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西十三宿舍"},
        {203, 220, {288, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
```

```
20000, 20000}, 1, "西十四宿舍"},
        {220, 219, {286, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西十五宿舍"},
        {219, 210, {286, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西十六宿舍"},
        {220, 201, {291, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西十七宿舍"},
        {436, 323, {218, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "南一宿舍"},
        {435, 335, {217, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "南二宿舍"},
        {470, 339, {215, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "南三宿舍"},
        {578, 285, {392, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东一宿舍"},
        {549, 240, {163, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东二宿舍"},
        {523, 252, {205, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东三宿舍"},
        {523, 239, {161, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东四宿舍"},
        {473, 238, {201, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东五宿舍"},
        {580, 268, {174, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东六宿舍"},
        {581, 254, {171, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东七宿舍"},
        {548, 253, {203, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东八宿舍"},
        {584, 301, {180, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东九宿舍"},
        {584, 312, {183, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东十宿舍"},
```

```
        {584, 322, {186, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东十一宿舍"},
        {583, 333, {189, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东十二宿舍"},
        {583, 344, {192, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "东十三宿舍"},
        {884, 238, {148, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 1 栋"},//
        {884, 255, {145, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 2 栋"},//
        {883, 212, {142, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 3 栋"},//
        {884, 198, {136, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 4 栋"},
        {891, 139, {130, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 5 栋"},
        {857, 128, {362, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 6 栋"},
        {875, 128, {110, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 7 栋"},
        {859, 117, {361, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 8 栋"},
        {875, 116, {105, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 9 栋"},
        {876, 107, {105, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 10 栋"},
        {891, 107, {104, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 11 栋"},
        {867, 95, {107, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 12 栋"},
        {908, 151, {364, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "韵苑 13 栋"},
        {935, 151, {124, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
```

20000, 20000}, 1, "韵苑 14 栋"},
{970, 152, {119, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 15 栋"},
{915, 139, {364, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 16 栋"},
{937, 139, {115, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 17 栋"},
{972, 139, {119, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 18 栋"},
{988, 139, {363, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 19 栋"},
{914, 129, {112, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 20 栋"},
{905, 119, {168, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 21 栋"},
{919, 109, {393, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 22 栋"},
{918, 99, {394, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 23 栋"},
{890, 94, {396, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 24 栋"},
{909, 87, {395, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 25 栋"},
{891, 84, {360, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 26 栋"},
{869, 83, {359, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 27 栋"},
{876, 88, {359, 398, 0, 0, 0, 0}, {25, 59, 20000, 20000, 20000, 20000}, 2, "99 号路口"},
{900, 89, {101, 360, 0, 0, 0, 0}, {14, 28, 20000, 20000, 20000, 20000}, 2, "100 号路口"},
{901, 93, {100, 395, 397, 0, 0, 0}, {14, 25, 23, 20000, 20000, 20000}, 3, "101 号路口"},

```
        {903, 100, {101, 396, 397, 0, 0, 0}, {25, 50, 15, 20000, 20000,
20000}, 3, "102 号路口"},
        {910, 113, {102, 114, 168, 393, 0, 0}, {45, 50, 33, 25, 20000,
20000}, 4, "103 号路口"},
        {891, 113, {82, 105, 168, 0, 0, 0}, {20, 56, 41, 20000, 20000,
20000}, 3, "104 号路口"},
        {875, 112, {81, 104, 106, 0, 0, 0}, {20, 56, 26, 20000, 20000,
20000}, 3, "105 号路口"},
        {868, 112, {105, 107, 108, 0, 0, 0}, {26, 39, 38, 20000, 20000,
20000}, 3, "106 号路口"},
        {868, 100, {83, 398, 106, 0, 0, 0}, {20, 25, 106, 20000, 20000,
20000}, 3, "107 号路口"},
        {867, 112, {104, 109, 361, 0, 0, 0}, {38, 40, 55, 20000, 20000,
20000}, 3, "108 号路口"},
        {866, 112, {108, 110, 130, 362, 0, 0}, {40, 28, 42, 26, 20000,
20000}, 4, "109 号路口"},
         {876, 133, {109, 111, 0, 0, 0, 0}, {28, 71, 20000, 20000,
20000, 20000}, 2, "110 号路口"},
         {895, 133, {110, 112, 0, 0, 0, 0}, {71, 32, 20000, 20000,
20000, 20000}, 2, "111 号路口"},
        {906, 134, {87, 111, 113, 0, 0, 0}, {20, 32, 60, 20000, 20000,
20000}, 3, "112 号路口"},
        {922, 135, {112, 114, 115, 125, 0, 0}, {60, 40, 70, 38, 20000,
20000}, 4, "113 号路口"},
        {922, 124, {103, 113, 115, 0, 0, 0}, {50, 40, 82, 20000, 20000,
20000}, 3, "114 号路口"},
        {943, 131, {88, 114, 116, 0, 0, 0}, {20, 82, 26, 20000, 20000,
20000}, 3, "115 号路口"},
        {950, 135, {115, 117, 123, 0, 0, 0}, {26, 38, 42, 20000, 20000,
20000}, 3, "116 号路口"},
         {961, 134, {116, 118, 0, 0, 0, 0}, {38, 42, 20000, 20000,
20000, 20000}, 2, "117 号路口"},
        {962, 145, {117, 119, 120, 0, 0, 0}, {42, 26, 39, 20000, 20000,
```

20000}, 3, "118 号路口"},
{970, 145, {86, 89, 118, 363, 0, 0}, {20, 20, 26, 66, 20000, 20000}, 4, "119 号路口"},
{961, 157, {118, 121, 406, 0, 0, 0}, {39, 19, 24, 20000, 20000, 20000}, 3, "120 号路口"},
{956, 157, {120, 122, 140, 0, 0, 0}, {19, 20, 172, 20000, 20000, 20000}, 3, "121 号路口"},
{951, 157, {121, 123, 126, 0, 0, 0}, {20, 37, 103, 20000, 20000, 20000}, 3, "122 号路口"},
{950, 145, {116, 122, 124, 0, 0, 0}, {42, 37, 48, 20000, 20000, 20000}, 3, "123 号路口"},
{936, 146, {85, 123, 125, 0, 0, 0}, {20, 48, 55, 20000, 20000, 20000}, 3, "124 号路口"},
{922, 146, {113, 124, 126, 364, 0, 0}, {38, 55, 39, 73, 20000, 20000}, 4, "125 号路口"},
{922, 157, {122, 125, 127, 138, 0, 0}, {103, 39, 102, 172, 20000, 20000}, 4, "126 号路口"},
{895, 156, {111, 126, 128, 0, 0, 0}, {78, 102, 100, 20000, 20000, 20000}, 3, "127 号路口"},
{870, 156, {127, 129, 131, 135, 0, 0}, {100, 28, 61, 168, 20000, 20000}, 4, "128 号路口"},
{866, 156, {128, 130, 131, 0, 0, 0}, {28, 21, 60, 20000, 20000, 20000}, 3, "129 号路口"},
{866, 144, {18, 76, 109, 129, 0, 0}, {20, 20, 42, 21, 20000, 20000}, 4, "130 号路口"},
{852, 156, {1, 128, 129, 132, 0, 0}, {20, 61, 60, 28, 20000, 20000}, 4, "131 号路口"},
{845, 156, {131, 133, 134, 0, 0, 0}, {28, 27, 26, 20000, 20000, 20000}, 3, "132 号路口"},
{837, 156, {132, 134, 154, 0, 0, 0}, {27, 86, 461, 20000, 20000, 20000}, 3, "133 号路口"},
{845, 133, {132, 133, 362, 0, 0, 0}, {26, 86, 31, 20000, 20000, 20000}, 3, "134 号路口"},

{869, 205, {128, 136, 141, 0, 0, 0}, {168, 51, 50, 20000, 20000, 20000}, 3, "135 号路口"},
{883, 204, {75, 135, 137, 0, 0, 0}, {20, 51, 46, 20000, 20000, 20000}, 3, "136 号路口"},
{897, 205, {136, 138, 143, 0, 0, 0}, {46, 84, 53, 20000, 20000, 20000}, 3, "137 号路口"},
{922, 205, {126, 137, 139, 0, 0, 0}, {172, 84, 46, 20000, 20000, 20000}, 3, "138 号路口"},
{934, 205, {138, 140, 150, 0, 0, 0}, {46, 76, 242, 20000, 20000, 20000}, 3, "139 号路口"},
{957, 205, {121, 139, 0, 0, 0, 0}, {172, 76, 20000, 20000, 20000, 20000}, 2, "140 号路口"},
{868, 218, {135, 142, 365, 0, 0, 0}, {50, 51, 23, 20000, 20000, 20000}, 3, "141 号路口"},
{883, 220, {74, 141, 143, 0, 0, 0}, {20, 51, 45, 20000, 20000, 20000}, 3, "142 号路口"},
{897, 220, {137, 142, 146, 0, 0, 0}, {53, 45, 46, 20000, 20000, 20000}, 3, "143 号路口"},
{868, 231, {145, 365, 366, 0, 0, 0}, {52, 22, 25, 20000, 20000, 20000}, 3, "144 号路口"},
{883, 231, {73, 144, 146, 0, 0, 0}, {20, 52, 50, 20000, 20000, 20000}, 3, "145 号路口"},
{897, 232, {143, 145, 149, 0, 0, 0}, {46, 50, 41, 20000, 20000, 20000}, 3, "146 号路口"},
{868, 244, {148, 152, 366, 0, 0, 0}, {50, 112, 20, 20000, 20000, 20000}, 3, "147 号路口"},
{885, 244, {72, 147, 149, 0, 0, 0}, {20, 50, 50, 20000, 20000, 20000}, 3, "148 号路口"},
{896, 244, {146, 148, 0, 0, 0, 0}, {41, 50, 20000, 20000, 20000, 20000}, 2, "149 号路口"},
{934, 275, {139, 151, 0, 0, 0, 0}, {242, 128, 20000, 20000, 20000, 20000}, 2, "150 号路口"},
{901, 291, {150, 152, 0, 0, 0, 0}, {128, 117, 20000, 20000,

20000, 20000}, 2, "151 号路口"},
{868, 291, {147, 151, 153, 0, 0, 0}, {112, 117, 324, 20000, 20000, 20000}, 3, "152 号路口"},
{865, 384, {5, 152, 0, 0, 0, 0}, {20, 324, 20000, 20000, 20000, 20000}, 2, "153 号路口"},
{706, 148, {19, 133, 155, 0, 0, 0}, {20, 461, 508, 20000, 20000, 20000}, 3, "154 号路口"},
{560, 141, {154, 156, 367, 0, 0, 0}, {508, 170, 134, 20000, 20000, 20000}, 3, "155 号路口"},
{512, 140, {155, 157, 252, 0, 0, 0}, {170, 111, 355, 20000, 20000, 20000}, 3, "156 号路口"},
{511, 173, {156, 159, 247, 0, 0, 0}, {111, 106, 184, 20000, 20000, 20000}, 3, "157 号路口"},
{563, 203, {159, 164, 368, 0, 0, 0}, {185, 98, 48, 20000, 20000, 20000}, 3, "158 号路口"},
{511, 202, {157, 158, 160, 245, 0, 0}, {106, 185, 102, 179, 20000, 20000}, 4, "159 号路口"},
{511, 231, {159, 161, 206, 208, 0, 0}, {102, 50, 57, 188, 20000, 20000}, 4, "160 号路口"},
{524, 231, {62, 160, 162, 0, 0, 0}, {20, 50, 42, 20000, 20000, 20000}, 3, "161 号路口"},
{536, 231, {161, 163, 204, 0, 0, 0}, {42, 45, 57, 20000, 20000, 20000}, 3, "162 号路口"},
{547, 231, {60, 162, 164, 0, 0, 0}, {20, 45, 40, 20000, 20000, 20000}, 3, "163 号路口"},
{561, 231, {158, 163, 165, 169, 0, 0}, {98, 40, 115, 55, 20000, 20000}, 4, "164 号路口"},
{595, 232, {164, 166, 0, 0, 0, 0}, {115, 56, 20000, 20000, 20000, 20000}, 2, "165 号路口"},
{595, 249, {165, 169, 172, 0, 0, 0}, {56, 116, 50, 20000, 20000, 20000}, 3, "166 号路口"},
{300, 335, {237, 267, 0, 0, 0, 0}, {97, 89, 20000, 20000, 20000, 20000}, 2, "167 号路口"},

```
        {903, 113, {92, 103, 104, 0, 0, 0}, {20, 33, 41, 20000, 20000,
20000}, 3, "168 号路口"},
        {561, 247, {164, 166, 170, 203, 0, 0}, {55, 116, 53, 46, 20000,
20000}, 4, "169 号路口"},
        {562, 262, {169, 171, 175, 202, 0, 0}, {53, 59, 43, 46, 20000,
20000}, 4, "170 号路口"},
        {580, 262, {65, 170, 172, 0, 0, 0}, {20, 59, 59, 20000, 20000,
20000}, 3, "171 号路口"},
        {595, 262, {166, 171, 173, 0, 0, 0}, {50, 59, 43, 20000, 20000,
20000}, 3, "172 号路口"},
        {595, 276, {172, 174, 178, 0, 0, 0}, {43, 58, 67, 20000, 20000,
20000}, 3, "173 号路口"},
        {580, 276, {64, 173, 175, 0, 0, 0}, {20, 58, 58, 20000, 20000,
20000}, 3, "174 号路口"},
        {562, 275, {170, 174, 392, 0, 0, 0}, {43, 58, 45, 20000, 20000,
20000}, 3, "175 号路口"},
        {561, 293, {177, 194, 392, 0, 0, 0}, {38, 46, 20, 20000, 20000,
20000}, 3, "176 号路口"},
        {581, 294, {176, 178, 179, 0, 0, 0}, {38, 78, 42, 20000, 20000,
20000}, 3, "177 号路口"},
        {596, 295, {173, 177, 181, 0, 0, 0}, {67, 78, 33, 20000, 20000,
20000}, 3, "178 号路口"},
        {572, 307, {177, 180, 184, 0, 0, 0}, {42, 41, 37, 20000, 20000,
20000}, 3, "179 号路口"},
        {584, 307, {67, 179, 181, 0, 0, 0}, {20, 41, 41, 20000, 20000,
20000}, 3, "180 号路口"},
        {595, 307, {178, 180, 182, 0, 0, 0}, {33, 41, 39, 20000, 20000,
20000}, 3, "181 号路口"},
        {595, 317, {181, 183, 187, 0, 0, 0}, {39, 41, 36, 20000, 20000,
20000}, 3, "182 号路口"},
        {584, 317, {68, 182, 184, 0, 0, 0}, {20, 41, 41, 20000, 20000,
20000}, 3, "183 号路口"},
        {572, 317, {179, 183, 185, 0, 0, 0}, {37, 41, 37, 20000, 20000,
```

20000}, 3, "184 号路口"},
{571, 328, {184, 186, 190, 0, 0, 0}, {37, 41, 37, 20000, 20000, 20000}, 3, "185 号路口"},
{583, 328, {69, 185, 187, 0, 0, 0}, {20, 41, 41, 20000, 20000, 20000}, 3, "186 号路口"},
{595, 328, {182, 186, 188, 0, 0, 0}, {36, 41, 39, 20000, 20000, 20000}, 3, "187 号路口"},
{595, 339, {187, 189, 193, 0, 0, 0}, {39, 41, 36, 20000, 20000, 20000}, 3, "188 号路口"},
{583, 339, {70, 188, 190, 0, 0, 0}, {20, 41, 41, 20000, 20000, 20000}, 3, "189 号路口"},
{571, 339, {185, 189, 191, 0, 0, 0}, {37, 41, 34, 20000, 20000, 20000}, 3, "190 号路口"},
{571, 349, {190, 192, 0, 0, 0, 0}, {34, 41, 20000, 20000, 20000, 20000}, 2, "191 号路口"},
{583, 349, {71, 191, 193, 0, 0, 0}, {20, 41, 41, 20000, 20000, 20000}, 3, "192 号路口"},
{594, 349, {188, 192, 0, 0, 0, 0}, {36, 41, 20000, 20000, 20000, 20000}, 2, "193 号路口"},
{548, 294, {10, 176, 195, 0, 0, 0}, {20, 46, 53, 20000, 20000, 20000}, 3, "194 号路口"},
{533, 293, {9, 194, 196, 0, 0, 0}, {20, 53, 85, 20000, 20000, 20000}, 3, "195 号路口"},
{510, 294, {195, 197, 212, 0, 0, 0}, {85, 45, 72, 20000, 20000, 20000}, 3, "196 号路口"},
{510, 278, {8, 196, 198, 0, 0, 0}, {20, 45, 37, 20000, 20000, 20000}, 3, "197 号路口"},
{511, 269, {197, 199, 210, 0, 0, 0}, {37, 31, 177, 20000, 20000, 20000}, 3, "198 号路口"},
{510, 261, {198, 200, 206, 0, 0, 0}, {31, 87, 57, 20000, 20000, 20000}, 3, "199 号路口"},
{535, 261, {199, 202, 204, 0, 0, 0}, {87, 47, 51, 20000, 20000, 20000}, 3, "200 号路口"},

{473, 245, {63, 206, 207, 0, 0, 0}, {20, 133, 50, 20000, 20000, 20000}, 3, "201 号路口"},

{549, 261, {11, 170, 200, 0, 0, 0}, {20, 46, 47, 20000, 20000, 20000}, 3, "202 号路口"},

{548, 247, {66, 169, 204, 0, 0, 0}, {20, 46, 46, 20000, 20000, 20000}, 3, "203 号路口"},

{535, 247, {162, 200, 203, 205, 0, 0}, {57, 51, 46, 45, 20000, 20000}, 4, "204 号路口"},

{523, 246, {61, 204, 206, 0, 0, 0}, {20, 45, 45, 20000, 20000, 20000}, 3, "205 号路口"},

{511, 246, {160, 199, 201, 205, 0, 0}, {57, 57, 133, 45, 20000, 20000}, 4, "206 号路口"},

{458, 245, {201, 208, 209, 0, 0, 0}, {50, 57, 46, 20000, 20000, 20000}, 3, "207 号路口"},

{459, 229, {160, 207, 243, 0, 0, 0}, {188, 57, 245, 20000, 20000, 20000}, 3, "208 号路口"},

{458, 260, {207, 210, 227, 0, 0, 0}, {46, 30, 155, 20000, 20000, 20000}, 3, "209 号路口"},

{458, 269, {198, 209, 211, 0, 0, 0}, {177, 30, 85, 20000, 20000, 20000}, 3, "210 号路口"},

{458, 292, {210, 212, 214, 226, 0, 0}, {85, 109, 133, 153, 20000, 20000}, 4, "211 号路口"},

{489, 293, {196, 211, 213, 0, 0, 0}, {72, 109, 130, 20000, 20000, 20000}, 3, "212 号路口"},

{489, 329, {212, 214, 0, 0, 0, 0}, {130, 110, 20000, 20000, 20000, 20000}, 2, "213 号路口"},

{457, 329, {211, 213, 215, 0, 0, 0}, {133, 110, 34, 20000, 20000, 20000}, 3, "214 号路口"},

{457, 339, {58, 214, 216, 0, 0, 0}, {20, 34, 20, 20000, 20000, 20000}, 3, "215 号路口"},

{457, 345, {215, 217, 219, 0, 0, 0}, {20, 62, 78, 20000, 20000, 20000}, 3, "216 号路口"},

{438, 344, {57, 216, 220, 223, 0, 0}, {20, 62, 78, 89, 20000,

20000}, 4, "217 号路口"},
{435, 329, {56, 214, 224, 0, 0, 0}, {20, 75, 76, 20000, 20000, 20000}, 3, "218 号路口"},
{458, 366, {216, 220, 0, 0, 0, 0}, {78, 66, 20000, 20000, 20000, 20000}, 2, "219 号路口"},
{438, 367, {217, 219, 221, 0, 0, 0}, {78, 66, 80, 20000, 20000, 20000}, 3, "220 号路口"},
{413, 367, {220, 222, 0, 0, 0, 0}, {80, 48, 20000, 20000, 20000, 20000}, 2, "221 号路口"},
{413, 353, {221, 223, 232, 0, 0, 0}, {48, 31, 33, 20000, 20000, 20000}, 3, "222 号路口"},
{414, 343, {217, 222, 224, 0, 0, 0}, {89, 31, 53, 20000, 20000, 20000}, 3, "223 号路口"},
{414, 329, {218, 223, 225, 0, 0, 0}, {76, 53, 30, 20000, 20000, 20000}, 3, "224 号路口"},
{414, 319, {224, 226, 233, 0, 0, 0}, {30, 101, 33, 20000, 20000, 20000}, 3, "225 号路口"},
{414, 292, {211, 225, 227, 230, 0, 0}, {153, 101, 108, 66, 20000, 20000}, 4, "226 号路口"},
{414, 259, {209, 226, 228, 0, 0, 0}, {155, 108, 91, 20000, 20000, 20000}, 3, "227 号路口"},
{390, 259, {227, 229, 242, 243, 0, 0}, {91, 109, 207, 108, 20000, 20000}, 4, "228 号路口"},
{390, 290, {228, 230, 240, 0, 0, 0}, {109, 23, 76, 20000, 20000, 20000}, 3, "229 号路口"},
{395, 290, {226, 229, 231, 0, 0, 0}, {66, 23, 100, 20000, 20000, 20000}, 3, "230 号路口"},
{396, 319, {230, 233, 239, 0, 0, 0}, {100, 35, 100, 20000, 20000, 20000}, 3, "231 号路口"},
{404, 353, {222, 233, 234, 235, 0, 0}, {33, 116, 131, 177, 20000, 20000}, 4, "232 号路口"},
{404, 320, {225, 231, 232, 0, 0, 0}, {33, 35, 116, 20000, 20000, 20000}, 3, "233 号路口"},

{367, 352, {232, 235, 236, 0, 0, 0}, {131, 140, 124, 20000, 20000, 20000}, 3, "234 号路口"},

{367, 393, {232, 234, 236, 0, 0, 0}, {177, 140, 177, 20000, 20000, 20000}, 3, "235 号路口"},

{329, 353, {234, 235, 237, 270, 0, 0}, {124, 177, 58, 52, 20000, 20000}, 4, "236 号路口"},

{331, 335, {167, 236, 238, 0, 0, 0}, {97, 58, 57, 20000, 20000, 20000}, 3, "237 号路口"},

{331, 318, {237, 239, 241, 264, 0, 0}, {57, 130, 97, 43, 20000, 20000}, 4, "238 号路口"},

{368, 320, {231, 238, 240, 0, 0, 0}, {100, 130, 101, 20000, 20000, 20000}, 3, "239 号路口"},

{367, 290, {229, 239, 241, 0, 0, 0}, {76, 101, 127, 20000, 20000, 20000}, 3, "240 号路口"},

{332, 290, {238, 240, 242, 262, 0, 0}, {97, 127, 108, 90, 20000, 20000}, 4, "241 号路口"},

{332, 259, {228, 241, 244, 0, 0, 0}, {207, 108, 116, 20000, 20000, 20000}, 3, "242 号路口"},

{390, 228, {208, 228, 244, 248, 0, 0}, {245, 108, 201, 100, 20000, 20000}, 4, "243 号路口"},

{332, 228, {242, 243, 251, 259, 0, 0}, {102, 201, 102, 117, 20000, 20000}, 4, "244 号路口"},

{459, 201, {159, 208, 246, 248, 0, 0}, {179, 97, 81, 243, 20000, 20000}, 4, "245 号路口"},

{460, 178, {245, 247, 0, 0, 0, 0}, {81, 18, 20000, 20000, 20000, 20000}, 2, "246 号路口"},

{460, 172, {157, 246, 249, 252, 0, 0}, {184, 18, 243, 113, 20000, 20000}, 4, "247 号路口"},

{391, 199, {243, 245, 249, 251, 0, 0}, {100, 243, 102, 204, 20000, 20000}, 4, "248 号路口"},

{391, 170, {247, 248, 250, 253, 0, 0}, {243, 102, 206, 117, 20000, 20000}, 4, "249 号路口"},

{331, 169, {249, 251, 255, 257, 0, 0}, {206, 102, 114, 180,

20000, 20000}, 4, "250 号路口"},
{333, 198, {244, 248, 250, 258, 0, 0}, {102, 204, 102, 186, 20000, 20000}, 4, "251 号路口"},
{460, 138, {156, 247, 253, 0, 0, 0}, {355, 113, 241, 20000, 20000, 20000}, 3, "252 号路口"},
{392, 137, {249, 252, 254, 384, 0, 0}, {117, 241, 87, 38, 20000, 20000}, 4, "253 号路口"},
{392, 112, {16, 253, 0, 0, 0, 0}, {20, 87, 20000, 20000, 20000, 20000}, 2, "254 号路口"},
{333, 135, {250, 256, 384, 0, 0, 0}, {114, 184, 208, 20000, 20000, 20000}, 3, "255 号路口"},
{232, 136, {255, 257, 311, 0, 0, 0}, {184, 115, 182, 20000, 20000, 20000}, 3, "256 号路口"},
{282, 168, {250, 256, 258, 309, 0, 0}, {180, 115, 102, 183, 20000, 20000}, 4, "257 号路口"},
{281, 198, {251, 257, 259, 295, 0, 0}, {186, 102, 103, 181, 20000, 20000}, 4, "258 号路口"},
{281, 228, {244, 258, 260, 280, 376, 399}, {117, 103, 218, 135, 82, 84}, 6, "259 号路口"},
{280, 290, {259, 261, 266, 281, 0, 0}, {218, 23, 99, 58, 20000, 20000}, 4, "260 号路口"},
{289, 289, {260, 262, 263, 0, 0, 0}, {23, 71, 65, 20000, 20000, 20000}, 3, "261 号路口"},
{308, 288, {241, 261, 263, 0, 0, 0}, {90, 71, 45, 20000, 20000, 20000}, 3, "262 号路口"},
{301, 301, {261, 262, 264, 265, 0, 0}, {65, 45, 93, 65, 20000, 20000}, 4, "263 号路口"},
{322, 319, {238, 263, 265, 0, 0, 0}, {43, 93, 122, 20000, 20000, 20000}, 3, "264 号路口"},
{293, 316, {263, 264, 266, 0, 0, 0}, {65, 122, 25, 20000, 20000, 20000}, 3, "265 号路口"},
{280, 289, {260, 265, 267, 0, 0, 0}, {99, 25, 40, 20000, 20000, 20000}, 3, "266 号路口"},

{280, 304, {167, 266, 268, 273, 0, 0}, {89, 40, 80, 210, 20000, 20000}, 4, "267 号路口"},
{280, 317, {267, 270, 271, 0, 0, 0}, {80, 130, 66, 20000, 20000, 20000}, 3, "268 号路口"},
{279, 379, {271, 369, 0, 0, 0, 0}, {35, 71, 20000, 20000, 20000, 20000}, 2, "269 号路口"},
{316, 352, {236, 268, 371, 0, 0, 0}, {52, 130, 33, 20000, 20000, 20000}, 3, "270 号路口"},
{279, 368, {268, 269, 272, 0, 0, 0}, {66, 35, 161, 20000, 20000, 20000}, 3, "271 号路口"},
{232, 381, {271, 273, 400, 0, 0, 0}, {161, 141, 12, 20000, 20000, 20000}, 3, "272 号路口"},
{223, 344, {267, 272, 275, 276, 0, 0}, {210, 141, 230, 134, 20000, 20000}, 4, "273 号路口"},
{170, 400, {275, 400, 0, 0, 0, 0}, {141, 230, 20000, 20000, 20000, 20000}, 2, "274 号路口"},
{159, 361, {273, 274, 277, 0, 0, 0}, {230, 141, 138, 20000, 20000, 20000}, 3, "275 号路口"},
{214, 307, {273, 277, 278, 281, 0, 0}, {134, 235, 86, 241, 20000, 20000}, 4, "276 号路口"},
{148, 323, {275, 276, 279, 0, 0, 0}, {138, 235, 118, 20000, 20000, 20000}, 3, "277 号路口"},
{207, 281, {276, 372, 0, 0, 0, 0}, {86, 25, 20000, 20000, 20000, 20000}, 2, "278 号路口"},
{144, 298, {277, 327, 0, 0, 0, 0}, {118, 30, 20000, 20000, 20000, 20000}, 2, "279 号路口"},
{251, 252, {259, 373, 374, 375, 0, 0}, {135, 62, 48, 55, 20000, 20000}, 4, "280 号路口"},
{264, 293, {260, 276, 374, 0, 0, 0}, {58, 241, 106, 20000, 20000, 20000}, 3, "281 号路口"},
{231, 224, {283, 285, 375, 377, 0, 0}, {64, 35, 69, 45, 20000, 20000}, 4, "282 号路口"},
{211, 225, {43, 282, 284, 287, 0, 0}, {20, 64, 66, 36, 20000,

20000}, 4, "283 号路口"},
{193, 225, {283, 289, 326, 372, 0, 0}, {66, 37, 58, 184, 20000, 20000}, 4, "284 号路口"},
{230, 215, {282, 286, 290, 0, 0, 0}, {35, 33, 27, 20000, 20000, 20000}, 3, "285 号路口"},
{219, 215, {53, 54, 285, 287, 0, 0}, {20, 20, 33, 33, 20000, 20000}, 4, "286 号路口"},
{211, 215, {283, 286, 288, 292, 0, 0}, {36, 33, 36, 37, 20000, 20000}, 4, "287 号路口"},
{203, 214, {51, 52, 287, 289, 0, 0}, {20, 20, 36, 35, 20000, 20000}, 4, "288 号路口"},
{190, 214, {284, 288, 294, 0, 0, 0}, {37, 35, 37, 20000, 20000, 20000}, 3, "289 号路口"},
{230, 207, {285, 401, 408, 0, 0, 0}, {27, 12, 65, 20000, 20000, 20000}, 3, "290 号路口"},
{220, 204, {55, 292, 401, 0, 0, 0}, {20, 35, 34, 20000, 20000, 20000}, 3, "291 号路口"},
{211, 204, {287, 291, 293, 297, 0, 0}, {37, 35, 42, 27, 20000, 20000}, 4, "292 号路口"},
{201, 204, {50, 292, 294, 0, 0, 0}, {20, 42, 41, 20000, 20000, 20000}, 3, "293 号路口"},
{187, 203, {289, 293, 299, 0, 0, 0}, {37, 41, 30, 20000, 20000, 20000}, 3, "294 号路口"},
{230, 196, {258, 296, 300, 401, 0, 0}, {181, 40, 48, 37, 20000, 20000}, 4, "295 号路口"},
{220, 196, {46, 295, 297, 0, 0, 0}, {20, 40, 30, 20000, 20000, 20000}, 3, "296 号路口"},
{212, 196, {292, 296, 298, 0, 0, 0}, {27, 30, 35, 20000, 20000, 20000}, 3, "297 号路口"},
{201, 196, {297, 299, 402, 0, 0, 0}, {35, 56, 38, 20000, 20000, 20000}, 3, "298 号路口"},
{185, 196, {294, 298, 321, 388, 0, 0}, {30, 56, 83, 26, 20000, 20000}, 4, "299 号路口"},

```
        {231, 182, {295, 301, 309, 0, 0, 0}, {48, 66, 55, 20000, 20000,
20000}, 3, "300 号路口"},
        {211, 182, {47, 300, 302, 0, 0, 0}, {20, 66, 33, 20000, 20000,
20000}, 3, "301 号路口"},
        {201, 182, {301, 304, 402, 0, 0, 0}, {33, 17, 10, 20000, 20000,
20000}, 3, "302 号路口"},
        {195, 185, {48, 402, 0, 0, 0, 0}, {20, 47, 20000, 20000, 20000,
20000}, 2, "303 号路口"},
        {201, 178, {302, 305, 391, 0, 0, 0}, {17, 34, 35, 20000, 20000,
20000}, 3, "304 号路口"},
        {201, 167, {304, 308, 309, 0, 0, 0}, {34, 82, 103, 20000,
20000, 20000}, 3, "305 号路口"},
        {180, 177, {307, 308, 391, 0, 0, 0}, {17, 36, 36, 20000, 20000,
20000}, 3, "306 号路口"},
        {182, 182, {306, 317, 388, 0, 0, 0}, {17, 56, 24, 20000, 20000,
20000}, 3, "307 号路口"},
        {178, 167, {305, 306, 312, 387, 0, 0}, {82, 36, 116, 36, 20000,
20000}, 4, "308 号路口"},
        {231, 167, {257, 300, 305, 311, 0, 0}, {183, 55, 103, 113,
20000, 20000}, 4, "309 号路口"},
        {123, 192, {314, 315, 0, 0, 0, 0}, {40, 21, 20000, 20000,
20000, 20000}, 2, "310 号路口"},
        {232, 135, {256, 309, 0, 0, 0, 0}, {182, 113, 20000, 20000,
20000, 20000}, 2, "311 号路口"},
        {146, 175, {308, 313, 314, 0, 0, 0}, {115, 54, 92, 20000,
20000, 20000}, 3, "312 号路口"},
        {151, 190, {312, 316, 317, 320, 0, 0}, {54, 41, 56, 49, 20000,
20000}, 4, "313 号路口"},
        {121, 181, {310, 312, 0, 0, 0, 0}, {40, 92, 20000, 20000,
20000, 20000}, 2, "314 号路口"},
        {129, 196, {310, 316, 386, 0, 0, 0}, {21, 41, 34, 20000, 20000,
20000}, 3, "315 号路口"},
        {142, 192, {42, 313, 315, 0, 0, 0}, {20, 41, 41, 20000, 20000,
```

20000}, 3, "316 号路口"},
{157, 188, {40, 307, 313, 39, 0, 0}, {20, 56, 56, 20, 20000, 20000}, 4, "317 号路口"},
{133, 209, {319, 385, 386, 0, 0, 0}, {40, 50, 15, 20000, 20000, 20000}, 3, "318 号路口"},
{144, 206, {41, 318, 320, 0, 0, 0}, {20, 40, 35, 20000, 20000, 20000}, 3, "319 号路口"},
{154, 203, {313, 319, 321, 0, 0, 0}, {49, 35, 30, 20000, 20000, 20000}, 3, "320 号路口"},
{162, 202, {299, 320, 322, 0, 0, 0}, {83, 30, 62, 20000, 20000, 20000}, 3, "321 号路口"},
{166, 219, {321, 379, 390, 0, 0, 0}, {62, 24, 52, 20000, 20000, 20000}, 3, "322 号路口"},
{137, 225, {324, 385, 390, 0, 0, 0}, {40, 51, 52, 20000, 20000, 20000}, 3, "323 号路口"},
{141, 237, {323, 325, 326, 0, 0, 0}, {40, 43, 130, 20000, 20000, 20000}, 3, "324 号路口"},
{128, 241, {324, 327, 0, 0, 0, 0}, {43, 175, 20000, 20000, 20000, 20000}, 2, "325 号路口"},
{176, 228, {284, 324, 379, 0, 0, 0}, {58, 130, 61, 20000, 20000, 20000}, 3, "326 号路口"},
{140, 291, {279, 325, 328, 0, 0, 0}, {30, 175, 110, 20000, 20000, 20000}, 3, "327 号路口"},
{111, 296, {327, 329, 353, 0, 0, 0}, {110, 16, 15, 20000, 20000, 20000}, 3, "328 号路口"},
{111, 292, {17, 328, 330, 0, 0, 0}, {20, 16, 39, 20000, 20000, 20000}, 3, "329 号路口"},
{101, 291, {23, 329, 331, 381, 0, 0}, {20, 39, 21, 17, 20000, 20000}, 4, "330 号路口"},
{94, 291, {330, 332, 0, 0, 0, 0}, {21, 44, 20000, 20000, 20000, 20000}, 2, "331 号路口"},
{81, 290, {22, 26, 331, 333, 0, 0}, {20, 20, 44, 45, 20000, 20000}, 4, "332 号路口"},

```
        {68, 290, {332, 334, 351, 0, 0, 0}, {45, 47, 50, 20000, 20000,
20000}, 3, "333 号路口"},
        {55, 289, {21, 25, 333, 380, 0, 0}, {20, 20, 47, 42, 20000,
20000}, 4, "334 号路口"},
        {41, 296, {24, 336, 380, 0, 0, 0}, {20, 17, 23, 20000, 20000,
20000}, 3, "335 号路口"},
        {43, 301, {335, 337, 404, 0, 0, 0}, {17, 26, 25, 20000, 20000,
20000}, 3, "336 号路口"},
        {37, 305, {27, 336, 338, 0, 0, 0}, {20, 26, 20, 20000, 20000,
20000}, 3, "337 号路口"},
        {31, 307, {28, 337, 339, 0, 0, 0}, {20, 20, 20, 20000, 20000,
20000}, 3, "338 号路口"},
        {34, 314, {29, 338, 340, 403, 0, 0}, {20, 20, 27, 42, 20000,
20000}, 4, "339 号路口"},
        {26, 314, {339, 341, 0, 0, 0, 0}, {27, 41, 20000, 20000, 20000,
20000}, 2, "340 号路口"},
        {27, 327, {340, 342, 343, 0, 0, 0}, {41, 35, 35, 20000, 20000,
20000}, 3, "341 号路口"},
        {30, 336, {341, 344, 0, 0, 0, 0}, {35, 32, 20000, 20000, 20000,
20000}, 2, "342 号路口"},
        {37, 325, {29, 341, 345, 0, 0, 0}, {20, 35, 34, 20000, 20000,
20000}, 3, "343 号路口"},
        {39, 335, {32, 343, 346, 0, 0, 0}, {20, 32, 33, 20000, 20000,
20000}, 3, "344 号路口"},
        {46, 324, {343, 346, 347, 403, 0, 0}, {34, 36, 43, 40, 20000,
20000}, 4, "345 号路口"},
        {48, 333, {344, 345, 348, 0, 0, 0}, {33, 36, 45, 20000, 20000,
20000}, 3, "346 号路口"},
        {57, 320, {30, 345, 350, 0, 0, 0}, {20, 43, 43, 20000, 20000,
20000}, 3, "347 号路口"},
        {60, 330, {31, 346, 349, 0, 0, 0}, {20, 45, 45, 20000, 20000,
20000}, 3, "348 号路口"},
        {72, 328, {348, 350, 405, 0, 0, 0}, {45, 33, 90, 20000, 20000,
```

20000}, 3, "349 号路口"},
{71, 318, {347, 349, 351, 355, 0, 0}, {43, 33, 51, 56, 20000, 20000}, 4, "350 号路口"},
{68, 304, {333, 350, 352, 404, 0, 0}, {50, 51, 76, 84, 20000, 20000}, 4, "351 号路口"},
{95, 303, {351, 381, 389, 0, 0, 0}, {76, 29, 31, 20000, 20000, 20000}, 3, "352 号路口"},
{109, 301, {36, 328, 381, 383, 389, 0}, {20, 15, 28, 34, 31, 20000}, 5, "353 号路口"},
{95, 314, {355, 382, 405, 0, 0, 0}, {56, 30, 31, 20000, 20000, 20000}, 3, "354 号路口"},
{83, 316, {31, 34, 350, 354, 0, 0}, {20, 20, 56, 56, 20000, 20000}, 4, "355 号路口"},
{119, 318, {37, 357, 358, 0, 0, 0}, {20, 22, 29, 20000, 20000, 20000}, 3, "356 号路口"},
{112, 319, {38, 356, 405, 0, 0, 0}, {20, 22, 57, 20000, 20000, 20000}, 3, "357 号路口"},
{118, 310, {356, 383, 0, 0, 0, 0}, {29, 26, 20000, 20000, 20000, 20000}, 2, "358 号路口"},
{870, 89, {98, 99, 0, 0, 0, 0}, {20, 25, 20000, 20000, 20000, 20000}, 2, "359 号路口"},
{893, 91, {97, 100, 0, 0, 0, 0}, {20, 67, 20000, 20000, 20000, 20000}, 2, "360 号路口"},
{859, 122, {79, 108, 0, 0, 0, 0}, {20, 55, 20000, 20000, 20000, 20000}, 2, "361 号路口"},
{857, 132, {77, 109, 134, 0, 0, 0}, {20, 26, 31, 20000, 20000, 20000}, 3, "362 号路口"},
{986, 144, {90, 119, 0, 0, 0, 0}, {20, 66, 20000, 20000, 20000, 20000}, 2, "363 号路口"},
{910, 145, {84, 125, 0, 0, 0, 0}, {20, 73, 20000, 20000, 20000, 20000}, 2, "364 号路口"},
{868, 225, {4, 141, 144, 0, 0, 0}, {20, 23, 22, 20000, 20000, 20000}, 3, "365 号路口"},

{868, 238, {3, 144, 147, 0, 0, 0}, {20, 25, 20, 20000, 20000, 20000}, 3, "366 号路口"},
{560, 181, {155, 368, 0, 0, 0, 0}, {134, 35, 20000, 20000, 20000, 20000}, 2, "367 号路口"},
{563, 189, {158, 367, 0, 0, 0, 0}, {48, 35, 20000, 20000, 20000, 20000}, 2, "368 号路口"},
{301, 380, {269, 370, 0, 0, 0, 0}, {71, 41, 20000, 20000, 20000, 20000}, 2, "369 号路口"},
{308, 373, {369, 371, 0, 0, 0, 0}, {41, 45, 20000, 20000, 20000, 20000}, 2, "370 号路口"},
{309, 369, {270, 370, 0, 0, 0, 0}, {33, 45, 20000, 20000, 20000, 20000}, 2, "371 号路口"},
{205, 275, {278, 284, 373, 0, 0, 0}, {25, 184, 118, 20000, 20000, 20000}, 3, "372 号路口"},
{238, 263, {280, 372, 0, 0, 0, 0}, {62, 118, 20000, 20000, 20000, 20000}, 2, "373 号路口"},
{259, 264, {280, 281, 0, 0, 0, 0}, {48, 106, 20000, 20000, 20000, 20000}, 2, "374 号路口"},
{247, 238, {280, 282, 376, 0, 0, 0}, {55, 69, 50, 20000, 20000, 20000}, 3, "375 号路口"},
{258, 221, {259, 407, 0, 0, 0, 0}, {82, 20, 20000, 20000, 20000, 20000}, 2, "376 号路口"},
{243, 224, {282, 407, 0, 0, 0, 0}, {45, 33, 20000, 20000, 20000, 20000}, 2, "377 号路口"},
{245, 230, {375, 407, 0, 0, 0, 0}, {27, 34, 20000, 20000, 20000, 20000}, 2, "378 号路口"},
{173, 218, {322, 326, 0, 0, 0, 0}, {24, 61, 20000, 20000, 20000, 20000}, 2, "379 号路口"},
{43, 290, {334, 335, 0, 0, 0, 0}, {42, 23, 20000, 20000, 20000, 20000}, 2, "380 号路口"},
{101, 296, {330, 352, 353, 0, 0, 0}, {17, 29, 28, 20000, 20000, 20000}, 3, "381 号路口"},
{101, 312, {35, 354, 383, 389, 0, 0}, {20, 25, 30, 17, 20000,

20000}, 4, "382 号路口"},
{112, 310, {353, 358, 382, 0, 0, 0}, {34, 26, 30, 20000, 20000, 20000}, 3, "383 号路口"},
{382, 137, {6, 253, 255, 0, 0, 0}, {20, 38, 208, 20000, 20000, 20000}, 3, "384 号路口"},
{135, 220, {12, 13, 318, 323, 0, 0}, {20, 20, 50, 51, 20000, 20000}, 4, "385 号路口"},
{131, 202, {14, 315, 318, 0, 0, 0}, {20, 34, 15, 20000, 20000, 20000}, 3, "386 号路口"},
{175, 158, {15, 308, 0, 0, 0, 0}, {20, 36, 20000, 20000, 20000, 20000}, 2, "387 号路口"},
{183, 188, {20, 299, 307, 0, 0, 0}, {20, 26, 24, 20000, 20000, 20000}, 3, "388 号路口"},
{100, 307, {352, 353, 382, 0, 0, 0}, {31, 31, 17, 20000, 20000, 20000}, 3, "389 号路口"},
{155, 222, {44, 322, 323, 0, 0, 0}, {20, 52, 52, 20000, 20000, 20000}, 3, "390 号路口"},
{192, 177, {49, 304, 306, 0, 0, 0}, {20, 35, 36, 20000, 20000, 20000}, 3, "391 号路口"},
{561, 288, {59, 175, 176, 0, 0, 0}, {20, 45, 20, 20000, 20000, 20000}, 3, "392 号路口"},
{919, 113, {93, 103, 0, 0, 0, 0}, {20, 28, 20000, 20000, 20000, 20000}, 2, "393 号路口"},
{917, 103, {94, 397, 0, 0, 0, 0}, {20, 44, 20000, 20000, 20000, 20000}, 2, "394 号路口"},
{909, 93, {96, 101, 0, 0, 0, 0}, {20, 32, 20000, 20000, 20000, 20000}, 2, "395 号路口"},
{890, 100, {95, 102, 0, 0, 0, 0}, {20, 50, 20000, 20000, 20000, 20000}, 2, "396 号路口"},
{904, 103, {102, 103, 394, 0, 0, 0}, {15, 45, 44, 20000, 20000, 20000}, 3, "397 号路口"},
{876, 100, {99, 107, 0, 0, 0, 0}, {59, 25, 20000, 20000, 20000, 20000}, 2, "398 号路口"},

{267, 208, {259, 0, 0, 0, 0, 0}, {84, 20000, 20000, 20000, 20000, 20000}, 1, "399 号路口"},

{233, 384, {271, 272, 274, 408, 0, 0}, {161, 12, 230, 63, 20000, 20000}, 4, "400 号路口"},

{231, 204, {290, 291, 295, 0, 0, 0}, {12, 34, 37, 20000, 20000, 20000}, 3, "401 号路口"},

{201, 185, {298, 302, 303, 0, 0, 0}, {38, 10, 47, 20000, 20000, 20000}, 3, "402 号路口"},

{44, 313, {336, 339, 345, 404, 0, 0}, {34, 42, 40, 28, 20000, 20000}, 4, "403 号路口"},

{49, 305, {336, 351, 403, 0, 0, 0}, {25, 84, 28, 20000, 20000, 20000}, 3, "404 号路口"},

{97, 323, {349, 354, 357, 0, 0, 0}, {90, 51, 57, 20000, 20000, 20000}, 3, "405 号路口"},

{969, 158, {2, 120, 0, 0, 0, 0}, {20, 24, 20000, 20000, 20000, 20000}, 2, "406 号路口"},

{252, 222, {376, 377, 378, 0, 0, 0}, {20, 33, 34, 20000, 20000, 20000}, 3, "407 号路口"},

{249, 207, {45, 290, 399, 0, 0, 0}, {20, 65, 63, 20000, 20000, 20000}, 3, "408 号路口"},

{884, 238, {148, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 1 栋驿站"},//409

{884, 255, {145, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 2 栋驿站"},//410

{883, 212, {142, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "韵苑 3 栋驿站"},//411

{880, 383, {153, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "东教工驿站"},//412

{550, 269, {202, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "东四驿站"},//413

{33, 301, {337, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000, 20000, 20000}, 1, "紫菘 13 栋驿站"},//414

{106, 317, {382, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,

```
20000, 20000}, 1, "紫菘 1 栋驿站"},//415
        {195, 188, {303, 0, 0, 0, 0, 0}, {20, 20000, 20000, 20000,
20000, 20000}, 1, "西十驿站"},//416
    };

    RouteState route_state;

    int random_int(int min, int max)
    {
        return rand() % (max - min + 1) + min;
    }

    void route(AcceptedOrder acp_orders[], int n_orders)
    {
        char debug_buf[120];
        int start_index,next_index;
        // UserList UL = {0};
        // USER *currentUser;
        // ReadAllUser(&UL); // 读取用户列表
        //currentUser=&UL.elem[user_pos];// 获取当前用户信息
        // OrderList OL = {0};
        // FoodList FL = {0};
        // DeliverList DL = {0};
        // ReadAllDeliver(&DL); // 读取快递列表
        // ReadAllOrder(&OL); // 读取订单列表
        // ReadAllFood(&FL); // 读取食品列表
        mouse_off_arrow(&mouse);
        draw_route();
        mouse_on_arrow(mouse);
        srand(time(NULL));

        // 初始化路线状态
        memset(&route_state, 0, sizeof(RouteState));
```

```
        route_state.remaining = n_orders * 2;
        sprintf(debug_buf,"%d",route_state.remaining);
        PrintText(1,1,debug_buf,HEI,24,1,black);
        route_state.current_pos = random_int(1, 409);
          next_index  =  arrange(route_state.current_pos,  acp_orders,
n_orders); // 随机生成起点
        sprintf(debug_buf,"%d",next_index);
        PrintText(20,1,debug_buf,HEI,24,1,black);
        while(1)
        {
            mouse_show_arrow(&mouse);
            if(mouse_press(122, 50, 242, 100)==1) //返回
            {
                // DestroyOList(&OL); // 释放订单列表内存
                // DestroyFList(&FL); // 释放食品列表内存
                // DestroyDList(&DL); // 释放快递列表内存
                return;
                //business(users.pos);
            }
            else if(mouse_press(342, 50, 462, 100)==1)
            {
                press3(1);//进入接单界面
                accept_order();//接单页面
                //return 后从这开始
                mouse_off_arrow(&mouse);
                bar1(0, 150, 1024, 768, white); // 清除接单界面残留
                draw_rider();
                mouse_on_arrow(mouse);
            }
            else if(mouse_press(562, 50, 682, 100)==1) //路线
            {
                press3(2); //按钮高亮
                mouse_off_arrow(&mouse);
```

```
    bar1(0, 150, 1024, 768, white); // 清除接单界面残留
    route(acp_orders,delivers.acp_count);//骑手路线规划
    //return 后从这开始
    mouse_on_arrow(mouse);
    bar1(0, 150, 1024, 768, white); // 清除路线界面残留
    draw_route();
    mouse_on_arrow(mouse);
}
else if(mouse_press(782, 50, 902, 100)==1) //我的
{
    press3(3); //按钮高亮
    mouse_off_arrow(&mouse);
    my_accept_order();
    //return 后从这开始
    mouse_on_arrow(mouse);
    bar1(0, 150, 1024, 768, white); // 清除路线界面残留
    draw_route();
    mouse_on_arrow(mouse);
}
else if (mouse_press(900, 266, 1020, 316)) {
    mouse_off_arrow(&mouse);
    bar1(0, 150, 1024, 326, white); // 清除已完成界面残留
    Readbmp64k(0, 326, "bmp\\map4.bmp");
    if (route_state.remaining > 0)
    {
        route_state.remaining--;
        route_state.current_pos = route_state.next_pos;
        if(route_state.next_type == 0)
        {
            route_state.picked[next_index] = 1;
        }
        else
        {
```

```
                    route_state.delivered[next_index] = 1;
                }
                    next_index = arrange(route_state.current_pos,
acp_orders, n_orders);
                }
            }
        }
    }

    int dijkstra(Node *start, Node *end, int count)
    {
        int distance[MAX_NODES]; // 存储从起点到每个节点的最短距离
        int visited[MAX_NODES] = {0}; // 标记节点是否已访问
        int prev[MAX_NODES]; // 存储每个节点的前驱节点
        int i, j, u, v, min_dist, alt;
        int path[MAX_NODES];
        int path_len = 0;
        int current = end - node;
        char buffer[200];
        int y_offset=50;
        int x1, y1, x2, y2;
        // 初始化
        for (i = 0; i < MAX_NODES; i++) {
            distance[i] = INF;  // 起点到每个节点的最短路径为无穷大
            prev[i] = -1;       // 前驱节点初始化为 -1
        }
        distance[start - node] = 0; // 起点到自身的距离为 0

        // 每一轮 i 循环找到一个节点的最短路径
        for (i = 0; i < MAX_NODES; i++) {
            min_dist = INF;
            u = -1;
```

```
    // 找到未访问节点中距离最小的节点
    for (j = 0; j < MAX_NODES; j++) {
        if (!visited[j] && distance[j] < min_dist) {
            min_dist = distance[j];
            u = j; // 记录距离最小节点的索引
        }
    }

    if (u == -1) break; // 所有节点都已访问或不可达
    visited[u] = 1;

    // 更新相邻节点的距离
    for (j = 0; j < node[u].num_of_adj_nodes; j++) {
        v = node[u].adj_nodes[j];
        if (v == 0) continue; // 跳过无效节点
        alt = distance[u] + node[u].distance[j];
        if (alt < distance[v]) {
            distance[v] = alt;
            prev[v] = u;
        }
    }
}

// 回溯路径
while (current != -1 && path_len < MAX_NODES) {
    path[path_len++] = current;
    current = prev[current];
}



for (i = path_len - 1; i > 0; i--)
{
        x1 = node[path[i]].x;
```

```
            y1 = node[path[i]].y + 326; // 地图坐标偏移补偿
            x2 = node[path[i - 1]].x;
            y2 = node[path[i - 1]].y + 326;
            if(count == 1)
            Line2(x1, y1, x2, y2, Red);
        }

        return distance[end - node]; // 返回最短距离
    }

    void draw_route()
    {
        bar1(0, 0, 1024, 768,white);
        bar1(0, 0, 1024, 150,deepblue);
        Readbmp64k(0, 326, "bmp\\map4.bmp");


        Fill_Rounded_Rectangle(122, 50, 242, 100, 25,white);//填色
        Fill_Rounded_Rectangle(342, 50, 462, 100, 25,white);//填色
        Fill_Rounded_Rectangle(562, 50, 682, 100, 25,deepblue);//填色
        Fill_Rounded_Rectangle(782, 50, 902, 100, 25,white);//填色


        Draw_Rounded_Rectangle(122, 50, 242, 100, 25, 1,deepblue);//
返回
        Draw_Rounded_Rectangle(342, 50, 462, 100, 25, 1,deepblue);//
接单
       Draw_Rounded_Rectangle(562, 50, 682, 100, 25, 1,white);//路线
        Draw_Rounded_Rectangle(782, 50, 902, 100, 25, 1,deepblue);//
账户


        PrintCC(122+35, 65, "返回", HEI, 24, 1, deepblue);
```

```
        PrintCC(342+35, 65, "接单", HEI, 24, 1, deepblue);
        PrintCC(562+35, 65, "路线", HEI, 24, 1, white);
        PrintCC(782+35, 65, "账户", HEI, 24, 1, deepblue);

        PrintCC(250, 10, "当前账号类型为：骑手", HEI, 24, 1, deepblue);

    }
/*****************************************************************/
    #include <all_func.h>

    Button button [] = {
        {0, 0, 0, 0, 0, 0}, // 占位符
        //东区
        {1, 273, 364, 393, 414, 59, 1}, // 1 栋
        {1, 459, 364, 579, 414, 60, 2}, // 2 栋
        {1, 645, 364, 765, 414, 61, 3}, // 3 栋
        {1, 831, 364, 951, 414, 62, 4}, // 4 栋
        {1, 273, 464, 393, 514, 63, 5}, // 5 栋
        {1, 459, 464, 579, 514, 64, 6}, // 6 栋
        {1, 645, 464, 765, 514, 65, 7}, // 7 栋
        {1, 831, 464, 951, 514, 66, 8}, // 8 栋
        {1, 273, 564 ,393 ,614 ,67, 9},  //9 栋
        {1 ,459 ,564 ,579 ,614 ,68, 10},//10 栋
        {1 ,645 ,564 ,765 ,614 ,69, 11},//11 栋
        {1 ,831 ,564 ,951 ,614 ,70, 12},//12 栋
        {1 ,273 ,664 ,393 ,714 ,71, 13},//13 栋

        // ---------------------------------
        // 西区
        {2, 242, 364, 362, 414, 39, 1}, // 1 栋
        {2, 397, 364, 517, 414, 40, 2}, // 2 栋
        {2, 552, 364, 672, 414, 41, 3}, // 3 栋
        {2, 707, 364, 827, 414, 42, 4}, // 4 栋
```

```
{2, 862, 364, 982, 414, 43, 5}, // 5 栋
{2, 242, 464, 362, 514, 44, 6}, // 6 栋
{2, 397, 464, 517, 514, 45, 7}, // 7 栋
{2, 552, 464 ,672 ,514 ,46, 8}, //8 栋
{2 ,707 ,464 ,827 ,514 ,47, 9}, //9 栋
{2 ,862 ,464 ,982 ,514 ,48, 10},//10 栋
{2 ,242 ,564 ,362 ,614 ,49, 11},//11 栋
{2 ,397 ,564 ,517 ,614 ,50, 12},//12 栋
{2 ,552 ,564 ,672 ,614 ,51, 13},//13 栋
{2 ,707 ,564 ,827 ,614 ,52, 14},//14 栋
{2 ,862 ,564 ,982 ,614 ,53, 15},//15 栋
{2 ,242 ,664 ,362 ,714 ,54, 16},//16 栋
{2 ,397 ,664 ,517 ,714 ,55, 17}, //17 栋

// ---------------------------------
// 南区
{3, 273, 364, 393, 414, 56, 1}, // 1 栋
{3, 459, 364, 579, 414, 57, 2}, // 2 栋
{3, 645, 364, 765, 414, 58, 3}, // 3 栋

// ---------------------------------
// 紫菘
{4, 242, 364, 362, 414, 35, 1}, // 1 栋
{4, 397, 364, 517, 414, 23, 2}, // 2 栋
{4, 552, 364, 672, 414, 34, 3}, // 3 栋
{4, 707, 364, 827, 414, 31, 4}, // 4 栋
{4, 862, 364, 982, 414, 26, 5}, // 5 栋
{4, 242, 464, 362, 514, 22, 6}, // 6 栋
{4, 397, 464 ,517 ,514 ,33, 7}, // 7 栋
{4, 552 ,464 ,672 ,514 ,30, 8}, // 8 栋
{4, 707 ,464 ,827 ,514 ,25, 9}, // 9 栋
{4, 862 ,464 ,982 ,514 ,21, 10},// 10 栋
{4, 242 ,564 ,362 ,614 ,32, 11},// 11 栋
```

```
{4, 397 ,564 ,517 ,614 ,29, 12},// 12 栋
{4, 552 ,564 ,672 ,614 ,27, 13},// 13 栋
{4, 707 ,564 ,827 ,614 ,24, 14},// 14 栋
{4, 862 ,564 ,982 ,614 ,28, 15},// 15 栋
{4, 242 ,664 ,362 ,714 ,36, 16},// 16 栋
{4, 397 ,664 ,517 ,714 ,37, 17},// 17 栋
{4, 552 ,664 ,672 ,714 ,38, 18},// 18 栋

// --------------------------------
// 韵苑
{5, 232, 340, 332, 390, 72, 1}, // 1 栋
{5, 364, 340, 464, 390, 73, 2}, // 2 栋
{5, 496, 340, 596, 390, 74, 3}, // 3 栋
{5, 628, 340, 728, 390, 75, 4}, // 4 栋
{5, 760, 340, 860, 390, 76, 5}, // 5 栋
{5, 892, 340 ,992 ,390 ,77, 6}, //6 栋
{5 ,232 ,425 ,332 ,475 ,78, 7}, //7 栋
{5 ,364 ,425 ,464 ,475 ,79, 8}, //8 栋
{5 ,496 ,425 ,596 ,475 ,80, 9}, //9 栋
{5 ,628 ,425 ,728 ,475 ,81, 10}, //10 栋
{5 ,760 ,425 ,860 ,475 ,82, 11}, //11 栋
{5 ,892 ,425 ,992 ,475 ,83, 12}, //12 栋
{5 ,232 ,510 ,332 ,560 ,84, 13}, //13 栋
{5 ,364 ,510 ,464 ,560 ,85, 14}, //14 栋
{5 ,496 ,510 ,596 ,560 ,86, 15}, //15 栋
{5 ,628 ,510 ,728 ,560 ,87, 16}, //16 栋
{5 ,760 ,510 ,860 ,560 ,88, 17}, //17 栋
{5, 892, 510, 992, 560, 89, 18}, //18 栋
{5, 232, 595, 332, 645, 90, 19}, //19 栋
{5, 364, 595, 464, 645, 91, 20}, //20 栋
{5, 496, 595, 596, 645, 92, 21}, //21 栋
{5, 628, 595, 728, 645, 93, 22}, //22 栋
{5, 760, 595, 860, 645, 94, 23}, //23 栋
```

```
        {5, 892 ,595 ,992 ,645 ,95, 24}, //24 栋
        {5 ,232 ,680 ,332 ,730 ,96, 25}, //25 栋
        {5 ,364 ,680 ,464 ,730 ,97, 26}, //26 栋
        {5 ,496 ,680 ,596 ,730 ,98, 27}, //27 栋
    };

    void draw_button(int x)
    {
        int i;
        int deta_x1 = (x == 5 ? 30 : 40), deta_x2 = (x == 5 ? 23 : 35),
deta_y = 15;

        char buffer[20];
        bar1(230, 320, 995, 732, white); // 清除原有按钮
         //PrintText(200, 310, "----------------------------------",
HEI, 32, 1, black); // 分隔线

        for (i = 0; i < sizeof(button) / sizeof(button[0]); i++) {
            if (button[i].commmunity == x) {
                   Draw_Rounded_Rectangle(button[i].x1, button[i].y1,
button[i].x2, button[i].y2, 25, 1, deepblue);
                sprintf(buffer, "%d 栋", button[i].number);
                if(button[i].number < 9) {
                     PrintText(button[i].x1 + deta_x1, button[i].y1 +
deta_y, buffer, HEI, 24, 1, deepblue);
                }
                else  {
                     PrintText(button[i].x1 + deta_x2, button[i].y1 +
deta_y, buffer, HEI, 24, 1, deepblue);
                }
            }
        }
    }
```

```
    int press_button(int mx, int my, int cur_index, int cur_community)
{
        int i, new_index = -1;
        char buffer[20], test_buf[20];
        int deta_x;
        int is_cur_index_valid;
        // 条件 1: 检查 cur_index 有效性
          is_cur_index_valid = (cur_index >= 0 && cur_index <
sizeof(button)/sizeof(button[0]));

        for (i = 0; i < sizeof(button)/sizeof(button[0]); i++) {
            // 条件 2: 检查鼠标点击范围、社区匹配、非自身按钮
            if (mx > button[i].x1 && mx < button[i].x2 &&
                my > button[i].y1 && my < button[i].y2 &&
                button[i].commmunity == cur_community &&
                       (!is_cur_index_valid || button[i].number !=
button[cur_index].number))
            {
                new_index = i;

                // 消除旧高亮（仅当 cur_index 有效时）
                if (is_cur_index_valid) {
                    // 恢复旧按钮颜色和文字
                          Fill_Rounded_Rectangle(button[cur_index].x1,
button[cur_index].y1,
                                                 button[cur_index].x2,
button[cur_index].y2, 25, white);
                          Draw_Rounded_Rectangle(button[cur_index].x1,
button[cur_index].y1,
                                                 button[cur_index].x2,
button[cur_index].y2, 25, 1, deepblue);
                    // 计算文本偏移量（基于社区）
```

```
                    deta_x = (button[cur_index].commmunity == 5) ? 23 :
35;
                    if (button[cur_index].number < 9) {
                        deta_x = (button[cur_index].commmunity == 5) ?
30 : 40;
                    }
                    sprintf(buffer, "%d 栋", button[cur_index].number);
                            PrintText(button[cur_index].x1  +  deta_x,
button[cur_index].y1 + 15,
                                buffer, HEI, 24, 1, deepblue);
                }else {

                }

                // 设置新高亮
                    Fill_Rounded_Rectangle(button[i].x1, button[i].y1,
button[i].x2, button[i].y2, 25, deepblue);
                    Draw_Rounded_Rectangle(button[i].x1, button[i].y1,
button[i].x2, button[i].y2, 25, 1, white);
                // 计算文本偏移量（基于社区）
                deta_x = (button[i].commmunity == 5) ? 23 : 35;
                if (button[i].number < 9) {
                    deta_x = (button[i].commmunity == 5) ? 30 : 40;
                }
                sprintf(buffer, "%d 栋", button[i].number);
                  PrintText(button[i].x1 + deta_x, button[i].y1 + 15,
buffer, HEI, 24, 1, white);

                break; // 退出循环，避免多个按钮被处理
            }
        }
        //调试信息
        // bar1(200, 310, 700, 340, white);
```

```
        //  sprintf(test_buf, "传出索引%d  对应%d  %d", new_index,
button[new_index].commmunity, button[new_index].number);
        // PrintText(200, 310, test_buf, HEI, 32, 1, black);

        return new_index;
    }
/*****************************************************************/
    #include "all_func.h"

    void de_order() {
        mouse_off_arrow(&mouse);
        draw_de_order();
        mouse_on_arrow(mouse);
        while (1) {
            mouse_show_arrow(&mouse);
            if(mouse_press(40, 113, 160, 163)==1)
            {
                return;
                //welcome();//首页
            }
            else if(mouse_press(40, 276, 160, 326)==1)
            {
                press_func(1);//进入超市页面
                user_shop();//用户超市页面
                return;
            }
            else if(mouse_press(40, 439, 160, 489)==1)
            {
                press_func(2);//进入外卖页面
                user_takeout();//用户外卖页面
                return;
            }
            else if(mouse_press(40, 602, 160, 652)==1)
```

```
            {
                press_func(3);//进入快递页面
                user_deliver();//用户快递页面
                return;
            }else if(mouse_press(800, 700, 1000, 750)==1)//确认并支付
            {
                 PrintCC(600, 715,"保存成功",HEI,24,1,lightred); // 支
付成功提示
                delay(500);
                bar1(600, 715,780,750,white); // 清除支付成功提示
                return;
            }
        }
    }

    void draw_de_order() {
        DeliverList DL={0};
        Deliver current; // 当前订单
        char time_str[100]; // 打印下单时间
        char user_name[100]; // 打印用户名
        char user_phone[100]; // 打印用户手机号
        char order_number[20]; // 打印订单号
        char address[100]; // 打印用户地址
        char community[50]; // 社区字符串
        char building[50]; // 楼栋字符串
        char code[20]; // 取件码字符串
        char company_str[20];// 快递公司字符串
        char station_str[20];// 站点字符串

        bar1(200, 0, 1024, 768,white);

        ReadAllDeliver(&DL); // 读取所有订单
```

```
        current = DL.elem[DL.length-1]; // 当前订单

        sprintf(order_number, "订单号：%d", current.id); // 订单号
        sprintf(time_str, "下单时间：%s",current.time );
        sprintf(user_name, "用户名：%s", current.name);
        sprintf(user_phone, "手机号：%s", current.number);
        sprintf(code, "取件码：%s", current.code);
         sprintf(company_str, "快递商：%s", companys[current.company-
1].name);
          sprintf(station_str, "站点：%s", stations[current.station-
1].name);

        if (current.station == 0)
                sprintf(address,"地址：未绑定地址");
            else
                sprintf(address, "地址：%s", node[current.index].name);
// 用户地址

        PrintText(250, 50, order_number, HEI, 24, 1, black);//显示订单
号
        PrintText(250, 100, time_str, HEI, 24, 1, black);//显示下单时
间
        PrintText(250, 150, user_name, HEI, 24, 1, black);//显示用户名
        PrintText(250, 200, user_phone, HEI, 24, 1, black);//显示手机
号
        PrintText(250, 250, address, HEI, 24, 1, black);//显示地址
        PrintText(250, 300, code, HEI, 24, 1, black);//显示取件码
        PrintText(250, 350, company_str, HEI, 24, 1, black);//显示快递
公司
        PrintText(250, 400, station_str, HEI, 24, 1, black);//显示站点

        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5, 1, deepblue);
// 确认并支付
```

```
        PrintCC(830, 715, "确认并支付", HEI, 24, 1, deepblue);

    }
/*****************************************************************/
    #include "all_func.h"

    FoodOrder Foodorders = {0}; // 订单

    void food_order(int index){

        UserList UL = {0};
        USER currentUser;

        int page = 0;// 初始页码
        int totalPage =(shopping_food.itemCount - 6 + 11 ) / 12 + 1 ;
// 总页数(向上取整)
        int state = 0; // 判断是否需要完善信息

        int cur_index = -1;
        int cur_community=0;
        int returned_index;

        float sum = 0.0; // 总金额

        ReadAllUser(&UL); // 读取用户列表

        currentUser=UL.elem[users.pos];// 获取当前用户信息

        DestroyUList(&UL); // 释放用户列表空间

        Foodorders.station = index; // 食堂编号

        mouse_off_arrow(&mouse);
```

```
        draw_food_order(page,&sum);

        mouse_on_arrow(mouse);

        while(1){
            mouse_show_arrow(&mouse);

            if (mouse_press(40, 113, 160, 163) == 1)
            {
                //
                return;
            }
            if(state==0)
            {
                if (mouse_press(220, 700, 340, 750) == 1)
                {
                    if (page > 0) {
                        page--;
                        draw_food_order(page,&sum);// 绘制用户订单页面
                    } else {
                        // 提示：已是第一页
                            PrintCC(550, 25, "已是第一页", HEI, 24, 1,
lightred);
                        delay(500);
                        bar1(550, 25, 700, 60, white);
                    }
                }
                else if (mouse_press(420, 700, 540, 750) == 1)
                {
                    if (page < totalPage - 1) {
                        page++;
                        draw_food_order(page,&sum);// 绘制用户订单页面
```

```
                    } else {
                        // 提示：已是最后一页
                          PrintCC(550, 25, "已是最后一页", HEI, 24, 1,
lightred);
                        delay(500);
                        bar1(550, 25, 700, 60, white);
                    }
                }
                else if(mouse_press(800, 700, 1000, 750) == 1)
                {
                             if  (currentUser.community  ==  '\0'  ||
strlen(currentUser.number) == 0)// 判断用户信息是否完善
                    {
                        draw_info();
                        state = 1;
                    }
                    else if(sum<10.0)
                    {
                         PrintText(700, 50, "未满 10 元，无法配送", HEI,
24, 1, lightred);
                        delay(500);
                        bar1(700, 50, 1024, 100, white);
                    }
                    else
                    {
                        save_food(Foodorders); // 保存订单
                           PrintCC(800, 50, "订单已保存", HEI, 24, 1,
lightred);
                        delay(500);
                        bar1(800, 50, 1024, 100, white);
                    }
                }
            }
```

```
            // 完善用户信息
            if(state==1)
            {
                if(mouse_press(430, 105, 650, 155)==1)
                {
                      number_input(currentUser.number, 435, 110, 645,
150); // 输入手机号
                }
                else if(mouse_press(710, 105, 830, 155)==1)
                {
                    if(strlen(currentUser.number)==11)
                    {
                        save_user(currentUser);
                                        strcpy(Foodorders.user_name,
currentUser.number);//保存手机号
                        save_food(Foodorders);//保存手机号到订单信息中
                        PrintCC(800,50,"保存成功",HEI,24,1,lightred);
                        delay(500);
                        bar1(800,50,950,100,snow);
                    }
                    else
                    {
                         PrintCC(800,50,"长度不合法",HEI,24,1,lightred);
                        delay(500);
                        bar1(800,50,950,100,snow);
                    }
                }
                else if(mouse_press(440, 180, 560, 230)==1)
                {
                    cur_index = -1;
                    press_func(4);//按钮状态切换
                    draw_button(1);
                    cur_community=1;
```

```
}
else if(mouse_press(620, 180, 740, 230)==1)
{
    cur_index = -1;
    press_func(5);//西区
    draw_button(2);
    cur_community=2;
}
else if(mouse_press(800, 180, 920, 230)==1)
{
    cur_index = -1;
    press_func(6);//南区
    draw_button(3);
    cur_community=3;

}
else if(mouse_press(530, 255, 650, 305)==1)
{
    cur_index = -1;
    press_func(7);//紫菘
    draw_button(4);
    cur_community=4;
}
else if(mouse_press(750, 255, 870, 305)==1)
{
    cur_index = -1;
    press_func(8);//韵苑
    draw_button(5);
    cur_community=5;
}
else if (mouse_press(200, 310, 1024, 768) == 1) {
```

```
                    MouseGet(&mouse);
                    mouse_off_arrow(&mouse);
                      returned_index = press_button(mouse.x, mouse.y,
cur_index, cur_community);//获取按钮编号

                     if(returned_index>=0)//如果返回值大于等于 0,则说明选
择了按钮
                    {
                                                currentUser.community   =
button[returned_index].commmunity;//获取社区编号
                                                    currentUser.index   =
button[returned_index].index;//获取楼号编号

                         Foodorders.community=currentUser.community;//
保存社区编号
                         Foodorders.station=currentUser.index;//保存楼
号编号

                        save_user(currentUser);//保存用户信息
                        save_food(Foodorders);//保存订单信息
                    }
                }

                if(mouse_press(950, 50, 975,75)==1)
                {
                    state = 0;
                    draw_food_order(page,&sum);
                }
            }
        }
    }

    void draw_food_order(int page,float *sum){
```

```
char address[100]; // 用户地址
int i;
UserList UL = {0};
FoodList FL = {0};
USER currentUser;
FoodOrder *currentFood;

char* current_time = get_current_time(); // 获取当前时间
char time_str[100]; // 打印下单时间
char user_name[100]; // 打印用户名
char user_phone[100]; // 打印用户手机号
char order_number; // 打印订单号

int startIdx = 0;// 起始商品索引
int itemsPerPage = 0;// 每页商品数量
int endIdx = 0;// 结束商品索引
int item_y = 0;// 商品框的 y 坐标

float total_amount = 0.0; // 总金额
char total_str[50]; // 总金额字符串
int fullPageItemCount = 0; // 满页商品数量

ReadAllUser(&UL); // 读取用户列表
currentUser = UL.elem[users.pos]; // 获取当前用户信息

ReadAllFood(&FL); // 读取订单列表
Foodorders.id = FL.length + 1; // 订单号

sprintf(time_str, "下单时间：%s", current_time);
sprintf(user_name, "用户名：%s", currentUser.name);
sprintf(user_phone, "手机号：%s", currentUser.number);

bar1(200, 0, 1024, 768, white); // 清空屏幕
```

```
        // 分页按钮
        Draw_Rounded_Rectangle(220, 700, 340, 750, 25, 1, deepblue);
// 上一页
        Draw_Rounded_Rectangle(420, 700, 540, 750, 25, 1, deepblue);
// 下一页
        PrintCC(245, 715, "上一页", HEI, 24, 1, deepblue);
        PrintCC(445, 715, "下一页", HEI, 24, 1, deepblue);

        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5, 1, deepblue);
// 确认并支付
        PrintCC(830, 715, "确认并支付", HEI, 24, 1, deepblue);

        // 页头信息只在第一页显示
        if (page == 0) {
            char order_number_str[20]; // 订单号字符串
            char community[50]; // 社区字符串
            char building[50]; // 楼栋字符串
            sprintf(order_number_str, "订单号：%d", Foodorders.id); //
订单号
            PrintText(250, 50, order_number_str, HEI, 24, 1, black);
            PrintText(250, 100, time_str, HEI, 24, 1, black);
            PrintText(250, 150, user_name, HEI, 24, 1, black);
            PrintText(250, 200, user_phone, HEI, 24, 1, black);
            sprintf(address, "地址：%s", node[currentUser.index].name);
// 用户地址
            PrintText(250, 250, address, HEI, 24, 1, black);

            PrintCC(750,250, canteen[Foodorders.station-1].name, HEI,
24, 1, black);//显示食堂名称

            // 表头
            PrintCC(250, 300, "商品详情：", HEI, 24, 1, black);
```

```
            PrintCC(750, 300, "数量：", HEI, 24, 1, black);
            PrintCC(900, 300, "金额：", HEI, 24, 1, black);
            PrintText(250, 320, "------------------------------", HEI,
32, 1, black);// 分隔线

            startIdx = 0;
            itemsPerPage = 6;
        } else {
            startIdx = 6 + (page - 1) * 12;
            itemsPerPage = 12;
        }

        endIdx = startIdx + itemsPerPage;
        if (endIdx > shopping_food.itemCount)// 防止越界
            endIdx = shopping_food.itemCount;

        item_y = (page == 0) ? 350 : 50;
        for (i = startIdx; i < endIdx; i++) {
            char total_str[50]; // 商品总价
            char quantity_str[20]; // 商品数量
            int productIndex = food_carts[i].index_in_foods; // 商品索
引
            int quantity = foods[productIndex].quantity;

              sprintf(total_str, "%.2f", foods[productIndex].price *
quantity);
            sprintf(quantity_str, "x%d", quantity);

               PrintCC(250, item_y, food_carts[i].name, HEI, 24, 1,
black); // 商品名
            PrintText(750, item_y, (unsigned char*)quantity_str, HEI,
24, 1, black);// 商品数量
            PrintText(900, item_y, (unsigned char*)total_str, HEI, 24,
```

```
1, black);// 商品金额

            item_y += 50;
        }

        // 判断是否需要在此页显示总金额（当前页没有满）
        fullPageItemCount = (page == 0) ? 6 : 12;// 第一页显示 6 个商品，其余页显示 12 个商品
                    if       ((endIdx       -       startIdx)       <
fullPageItemCount||endIdx==shopping_food.itemCount) {// 当前页商品数量不满一页或最后一个商品刚好满页都要打印出总金额
            //如果不是最后一个商品但是满页就不打印总金额
            // 打印分隔线
            PrintText(250, item_y - 30, "----------------------------
---", HEI, 32, 1, black);

            // 计算总金额
            total_amount = 0.0;
            for (i = 0; i < shopping_food.itemCount; i++) {
                int productIndex = food_carts[i].index_in_foods;
                int quantity = foods[productIndex].quantity;
                total_amount += foods[productIndex].price * quantity;
                 food_carts[i].quantity = quantity; // 记录购物车商品数量
                food_carts[i].price = foods[productIndex].price; // 记录商品价格
            }

            *sum=total_amount;

            sprintf(total_str, "总金额: %.2f 元", total_amount);
            PrintText(750, item_y + 10, total_str, HEI, 24, 1, black);
        }
```

```
        //存储订单信息
        strcpy(Foodorders.order_time, current_time); // 下单时间
        strcpy(Foodorders.user_name, currentUser.name); // 用户名
        strcpy(Foodorders.user_phone, currentUser.number); // 用户手机
号


        Foodorders.community=currentUser.community; // 用户社区
        Foodorders.building=currentUser.building;
        //Foodorders.station=; // 用户取餐地点
        Foodorders.destination=currentUser.index; // 用户送餐地点

        for (i = 0; i < shopping_food.itemCount; i++) {
            Foodorders.item[i] = food_carts[i]; // 购物车内商品信息
        }

         Foodorders.itemCount = shopping_food.itemCount; // 购物车内商
品数量
        Foodorders.total_amount = total_amount; // 总金额

        //食堂编号在 20 行存过了

        DestroyUList(&UL); // 释放用户列表空间
        DestroyFList(&FL); // 释放订单列表空间

    }

    /*****************************
    功能说明：得到元素在线性表中的位置
    参数说明：线性表，元素
    返回值：   如果存在就返回位置，否则返回-1
    *****************************/
```

```
int FoodOrder_pos(FoodList FL,FoodOrder Foodorders)
{
    int i=-1;
    for(i=0;i<FL.length;i++)
    {
        if(Foodorders.id == FL.elem[i].id)
        {
            return i;
        }
    }
    return -1;
}

/*****************************
功能说明：在线性表 L 末尾插入元素
参数说明：线性表地址，要插入的元素
返回值：  无
*****************************/
void FListInsert(FoodList*FL,FoodOrder Foodorders)
{
    FoodOrder*newbase=NULL;//创建新基址
    if(FL->length>=FL->listsize)//如果线性表已满
    {
        if((newbase=(FoodOrder*)realloc(FL->elem,(FL->listsize+F_
LISTINCEREMENT)*sizeof(FoodOrder)))==NULL)////重新分配内存
        {
        CloseSVGA();
        printf("No enough memory!\n");
        printf("FListInsert\n");
        exit(-1);
        }
        FL->elem=newbase;//更新基址
        FL->listsize+=F_LISTINCEREMENT;//更新线性表容量
```

```
    }
    *(FL->elem+(FL->length))=Foodorders;//插入新元素
    FL->length++;//线性表长度加一
}

/**********************
功能说明：保存账单信息函数
参数说明：账单结构体
返回值说明:0：保存成功    -1：  保存失败
**********************/
int save_food(FoodOrder Foodorders) {
    int i = 0;
    FoodList FL = {0};
    int order_pos;
    FILE *fp = NULL;

    ReadAllFood(&FL);//读取所有订单信息

    if ((fp = fopen("data\\Foodorder.dat", "wb")) == NULL) {
        printf("无法打开文件！\n");
        return -1;
    }

    // 先查找订单是否已经存在
    order_pos = FoodOrder_pos(FL, Foodorders);

    if (order_pos == -1)  // 如果订单不存在
    {
        FListInsert(&FL, Foodorders); // 插入订单
    }
    else  // 如果订单存在，更新原有订单信息
    {
        Foodorders.id = FL.elem[order_pos].id; // 保留原订单 ID
```

```
        FL.elem[order_pos] = Foodorders;
    }

    // 重新将线性表写入文件
    rewind(fp);//将文件指针移动到文件开头
    fwrite(&FL.length, sizeof(short), 1, fp);//写入线性表长度
    fwrite(&FL.listsize, sizeof(short), 1, fp);//写入线性表容量

    // 逐个写入数据
    for (i = 0; i < FL.length; i++) {
        fwrite(&FL.elem[i], sizeof(FoodOrder), 1, fp);
    }

    fclose(fp);
    DestroyFList(&FL);
    return 0;
}

void DestroyFList(FoodList*FL)
{
    free(FL->elem);
    FL->elem=NULL;
    FL->listsize=0;
    FL->length=0;
}

// 初始化线性表
void ReadAllFood(FoodList *FL) {
    int i = 0;
    short length = 0;//线性表初始长度
    short listsize = 10; //线性表初始容量（能够存储 10 个订单）
    FILE *fp = NULL;
```

```
        if ((fp = fopen("data\\Foodorder.dat", "rb")) == NULL) {//如果
打开文件失败
            fp = fopen("data\\Foodorder.dat", "wb");// 如果文件不存在则
创建一个新的文件
            if (fp == NULL) {//如果创建文件失败
                printf("无法创建文件！\n");
                return;//无法创建文件则返回，不需要继续执行下面的代码
            }
            fwrite(&length, sizeof(short), 1, fp); //如果创建成功则写入
初始长度 0
            fwrite(&listsize, sizeof(short), 1, fp); //写入初始容量 10
            fclose(fp);//关闭文件
            return;//创建完成后返回，不需要继续执行下面的代码
        }
        //如果打开文件成功则读取长度和容量
        fread(&length, sizeof(short), 1, fp);
        fread(&listsize, sizeof(short), 1, fp);
        //把读取的长度和容量赋值给线性表
        FL->length = length;
        FL->listsize = listsize;
        FL->elem = (FoodOrder *)malloc(listsize * sizeof(FoodOrder));
//分配存储空间
        //如果线性表的存储空间分配失败则输出错误信息并退出程序
        if (FL->elem == NULL) {
            printf("No enough memory!\n");
            printf("ReadAllOrder\n");
            fclose(fp);
            exit(-1);
        }
        //如果分配成功就逐个读取订单数据
        //并把读取的数据存储到线性表中
        for (i = 0; i < length; i++) {
            fread(&FL->elem[i], sizeof(FoodOrder), 1, fp);
```

```
        }

        fclose(fp);
    }
/*****************************************************************/
    #include "all_func.h"

    void user_cart() {
        int page = 0;// 初始页码
        int totalPage = (cart.itemCount + 3) / 4; // 向上取整
        float sum=0;//总价

        mouse_off_arrow(&mouse);

        draw_user_cart(carts, cart.itemCount, page,&sum);
        mouse_on_arrow(mouse);

        while (1) {
            mouse_show_arrow(&mouse);

            // 点击返回商店
            if (mouse_press(40, 113, 160, 163) == 1)
            {
                user_shop();
                return;
            }
            else if (mouse_press(800, 700, 1000, 750) == 1)
            {
                if(sum<10.0)
                {
                     PrintText(500, 25, "未满 10 元，无法配送", HEI, 24,
1, lightred);
                    delay(500);
```

```
            bar1(500, 25, 750, 60, white);
        }
        else
        {
            user_order();// 点击生成订单按钮，进入订单页面

            //return 后从这开始
            mouse_off_arrow(&mouse);
            bar1(200, 0, 1024, 768, white); // 清除注册界面残留
            draw_user_cart(carts, cart.itemCount, page,&sum);
            mouse_on_arrow(mouse);
        }

    }
    else if (mouse_press(220, 700, 340, 750) == 1)
    {
        if (page > 0) {
            page--;
            draw_user_cart(carts, cart.itemCount, page,&sum);
        } else {
            // 提示：已是第一页
            PrintCC(550, 25, "已是第一页", HEI, 24, 1, lightred);
            delay(500);
            bar1(550, 25, 700, 60, white);
        }
    }
    else if (mouse_press(420, 700, 540, 750) == 1)
    {
        if (page < totalPage - 1) {
            page++;
            draw_user_cart(carts, cart.itemCount, page,&sum);
        } else {
            // 提示：已是最后一页
```

```
                    PrintCC(550, 25, "已是最后一页", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(550, 25, 700, 60, white);
                }
            }
            else if(mouse_press(270, 0, 1024, 680) == 1) {
                MouseGet(&mouse);
                AddSub_cart(mouse.x, mouse.y, carts, &cart.itemCount,
page,&sum);
                delay(100);
            }

        }
    }

    void draw_user_cart(CartItem carts[], int cartCount, int
page,float *sum) {
        int i,k;
        int start = page * 4;// 起始商品索引
        int end = start + 4;// 结束商品索引
        char sum_str[20];//总价字符串
        if (end > cartCount) end = cartCount;// 防止越界

        bar1(200, 0, 1024, 768, white);
        bar1(0, 250, 199, 768, deepblue);

        Draw_Rounded_Rectangle(220, 700, 340, 750, 25, 1, deepblue);
// 上一页
        Draw_Rounded_Rectangle(420, 700, 540, 750, 25, 1, deepblue);
// 下一页
        PrintCC(245, 715, "上一页", HEI, 24, 1, deepblue);
        PrintCC(445, 715, "下一页", HEI, 24, 1, deepblue);
```

```
        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5, 1, deepblue);
// 生成订单
        PrintCC(850, 715, "生成订单", HEI, 24, 1, deepblue);

        for (i = start; i < end; i++) {//显示商品信息
            char total_str[50];//商品总价
            char quantity_str[20];//商品数量
            char type_str[20];//种类名
            int y = 10 + 170 * (i - start);//商品框的 y 坐标
            int productIndex = carts[i].index_in_products;//商品索引
            int quantity = products[productIndex].quantity;//商品数量

                        sprintf(total_str,    "   金   额   :%.2f",
products[productIndex].price * quantity);//将金额转换为字符串
            sprintf(quantity_str, "x%d", quantity);//将数量转换为字符串

             switch(products[productIndex].type){//根据商品类型显示种类
名
                case 1: strcpy(type_str, "生活用品"); break;
                case 2: strcpy(type_str, "文具"); break;
                case 3: strcpy(type_str, "零食"); break;
                case 4: strcpy(type_str, "饮料"); break;
                case 5: strcpy(type_str, "运动用品"); break;
                case 6: strcpy(type_str, "水果"); break;
                case 7: strcpy(type_str, "文创"); break;
                default: strcpy(type_str, "未知"); break;
            }

              Draw_Rounded_Rectangle(220, y, 1000, y + 150, 30, 1,
0x6B4D);//商品框
            Readbmp64k(240, y + 15, carts[i].photo);//显示商品图片
```

```
            PrintCC(370, y + 15, carts[i].name, HEI, 32, 1, 0x0000);//
显示商品名
            PrintCC(370, y + 110, type_str, HEI, 24, 1, 0x0000);//显示
种类名
            PrintText(760, y + 15, (unsigned char*)total_str, HEI, 32,
1, 0x0000);//显示金额
            PrintText(920, y + 60, (unsigned char*)quantity_str, HEI,
32, 1, 0x0000);//显示数量

            Line_Thick(840, y + 120, 860, y + 120, 1, black); // 减号

            Line_Thick(940, y + 120, 960, y + 120, 1, black); // 加号
横
            Line_Thick(950, y + 110, 950, y + 130, 1, black); // 加号
竖
        }

        *sum = 0;
        for (k = 0; k < cart.itemCount; k++) {
            int pIndex = carts[k].index_in_products;
            *sum += products[pIndex].price * products[pIndex].quantity;
        }//计算总价

        sprintf(sum_str, "总价:%.2f", *sum);
          PrintText(560,710, (unsigned char*)sum_str, HEI, 32, 1,
0x0000);//显示金额
    }

    // 更新显示商品数量
    void draw_user_cart_quantity(CartItem carts[], int index, int y)
{
        char total_str[50];
        char quantity_str[20];
```

```
        char sum_str[20];
        float sum = 0;
        int i;

        int productIndex = carts[index].index_in_products;
        int quantity = products[productIndex].quantity;

        sprintf(total_str, "金额:%.2f", products[productIndex].price *
quantity);
        sprintf(quantity_str, "x%d", quantity);

        // === 重新计算整个购物车总价 ===
        for (i = 0; i < cart.itemCount; i++) {
            int pIndex = carts[i].index_in_products;
            sum += products[pIndex].price * products[pIndex].quantity;
        }
        sprintf(sum_str, "总价:%.2f", sum);

        // === 清除原有数值显示区域 ===
        bar1(760, y + 15, 990, y + 60, white); // 金额区域
        bar1(920, y + 60, 990, y + 90, white); // 数量区域
        bar1(560, 710, 790, 750, white);       // 总价区域

        // === 显示新数值 ===
        PrintText(760, y + 15, (unsigned char*)total_str, HEI, 32, 1,
0x0000);     // 显示该商品金额
        PrintText(920, y + 60, (unsigned char*)quantity_str, HEI, 32,
1, 0x0000);  // 显示该商品数量
         PrintText(560, 710, (unsigned char*)sum_str, HEI, 32, 1,
0x0000);          // 显示整个购物车总价
    }

    // 添加或减少购物车中商品数量
```

```
    void AddSub_cart(int mx, int my, CartItem carts[], int* itemCount,
int currentPage,float *sum) {
        int i,k;
        int start = currentPage * 4;
        int end = start + 4;
        if (end > *itemCount) end = *itemCount;

        for (i = start; i < end; i++) {
            int localIndex = i - start;
            int y = 10 + 170 * localIndex;
             int productIndex = carts[i].index_in_products; // 映射回
products

            // 减号区域
            if (mx >= 840 && mx <= 860 && my >= y + 115 && my <= y +
125) {
                if (products[productIndex].quantity > 1) {
                    products[productIndex].quantity--;
                    // 重新计算总价
                    *sum = 0;
                    for (k = 0; k < cart.itemCount; k++)
                    {
                        int pIndex = carts[k].index_in_products;
                                    *sum  +=  products[pIndex].price  *
products[pIndex].quantity;
                    }
                     draw_user_cart_quantity(carts, i, y); // 仅更新该
商品
                } else {
                    int j;
                    // 移除商品
                    products[productIndex].quantity = 0;
```

```
                    // 从购物车中移除

                    for (j = i; j < *itemCount - 1; j++) {
                        carts[j] = carts[j + 1];
                    }
                    (*itemCount)--;

                    // 更新总价
                    *sum = 0;
                    for (k = 0; k < cart.itemCount; k++)
                    {
                        int pIndex = carts[k].index_in_products;
                                    *sum  +=  products[pIndex].price  *
products[pIndex].quantity;
                    }
                                    draw_user_cart(carts,  *itemCount,
currentPage,&sum); // 重绘整个页面
                }
                return;
            }

            // 加号区域
            if (mx >= 940 && mx <= 960 && my >= y + 115 && my <= y +
125) {
                products[productIndex].quantity++;
                for (k = 0; k < cart.itemCount; k++) {
                    int pIndex = carts[k].index_in_products;
                                *sum  +=  products[pIndex].price  *
products[pIndex].quantity;
                }
                draw_user_cart_quantity(carts, i, y); // 仅更新该商品
                return;
            }
```

```
        }
    }
/*****************************************************************/
    #include "all_func.h"

    Deliver delivers={0};//存储信息的快递结构体
    Company companys[8]={
        {"顺丰快递"},
        {"韵达快递"},
        {"申通快递"},
        {"中通快递"},
        {"京东快递"},
        {"邮政快递"},
        {"圆通快递"},
        {"其他快递"}
    };
    Station stations[8]={
        {"韵苑 1 栋"},
        {"韵苑 2 栋"},
        {"韵苑 3 栋"},
        {"东教工 17 栋"},
        {"东 4 驿站"},
        {"紫菘 13 栋"},
        {"紫菘 1 栋"},
        {"西十舍"},
    };

    void user_deliver(){
        UserList UL = {0};
        DeliverList DL = {0};
        USER currentUser;
        int last_index=-1;//记录上次选择的服务提供商
        int last_index_station=-1;//记录上次选择的站点
```

```
int state=0; //判断是否需要完善信息

int cur_index = -1;
int cur_community=0;
int returned_index;

ReadAllUser(&UL); // 读取用户列表
ReadAllDeliver(&DL); // 读取订单列表

currentUser=UL.elem[users.pos];// 获取当前用户信息

delivers.id=DL.length+1;// 订单号
strcpy(delivers.name, currentUser.name);// 用户名
strcpy(delivers.number, currentUser.number);// 用户手机号
delivers.community=currentUser.community;// 用户地址
delivers.building=currentUser.building;// 用户楼栋
delivers.index=currentUser.index;//

DestroyUList(&UL); // 释放用户列表空间

mouse_off_arrow(&mouse);

draw_user_deliver();

mouse_on_arrow(mouse);

while(1){
    mouse_show_arrow(&mouse);

    if(mouse_press(40, 113, 160, 163)==1)
    {
        return;
        //welcome();//首页
```

```
        }
        else if(mouse_press(40, 276, 160, 326)==1)
        {
            press_func(1);//进入超市页面
            user_shop();//用户超市页面
            return;
        }
        else if(mouse_press(40, 439, 160, 489)==1)
        {
            press_func(2);//进入外卖页面
            user_takeout();//用户外卖页面
            return;
        }
        else if(mouse_press(40, 602, 160, 652)==1)
        {
            press_func(3);//进入快递页面
            user_deliver();//用户快递页面
            return;
        }

        //
        if(state==0){
            if(mouse_press(440, 35, 660, 85)==1)
            {
                deliver_input(delivers.code, 445, 40, 655, 80); //
输入取件码
            }
             else if(mouse_press(730, 35, 850, 85)==1)//保存取件码
按钮
            {
                save_Deliver(delivers);
                PrintCC(750, 120, "保存成功", HEI, 24, 1, lightred);
                delay(500);
```

```
                bar1(750, 120, 1024, 160, white);
            }
            else if(mouse_press(800, 700, 1000, 750)==1)//点击生成
订单按钮
            {
                if (delivers.company == 0) // 未选择服务提供商
                {
                     PrintCC(750, 120, "请选择服务提供商", HEI, 24,
1, lightred);
                    delay(500);
                    bar1(750, 120, 1024, 160, white);
                }
                else if (delivers.station == 0) // 未选择站点
                {
                       PrintCC(750, 120, "请选择驿站", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(750, 120, 1024, 160, white);
                }
                else if(strlen(delivers.code)==0){//未输入取件码
                     PrintCC(750, 120, "请输入取件码", HEI, 24, 1,
lightred);
                    delay(500);
                    bar1(750, 120, 1024, 160, white);
                }
                          else if (currentUser.index == 0 ||
strlen(currentUser.number) == 0)// 判断用户信息是否完善
                {
                    mouse_off_arrow(&mouse);
                    draw_info();
                    mouse_on_arrow(mouse);
                    state = 1;
                }
```

```
                else//输入正确,保存信息
                {
                     strcpy(delivers.time, get_current_time()); //
保存时间
                    save_Deliver(delivers); // 保存订单信息
                    de_order();//进入订单页面
                }

            }
             else if(mouse_press(250, 175, 750+185, 375)==1)//选择
服务提供商
            {
                int index;
                MouseGet(&mouse);
                mouse_off_arrow(&mouse);

                          index=choose_company(mouse.x,  mouse.y,
&last_index);

                if(index!=-1)
                {
                    delivers.company=index;//记录选择的服务提供商
                    save_Deliver(delivers);
                }

                mouse_on_arrow(mouse);
            }
             else if(mouse_press(250, 455, 750+185, 655)==1)//选择
驿站
            {
                int index_station;
                MouseGet(&mouse);
                mouse_off_arrow(&mouse);
```

```
                    index_station=choose_station(mouse.x, mouse.y,
&last_index_station);

                if(index_station!=-1)
                {
                    delivers.station=index_station;//记录选择的站点
                    save_Deliver(delivers);
                }

                mouse_on_arrow(mouse);
            }
        }

        // 完善用户信息
        if(state==1)
        {
            if(mouse_press(430, 105, 650, 155)==1)
            {
                  number_input(currentUser.number, 435, 110, 645,
150); // 输入手机号
            }
            else if(mouse_press(710, 105, 830, 155)==1)
            {
                if(strlen(currentUser.number)==11)
                {
                    save_user(currentUser);
                    strcpy(delivers.number, currentUser.number);//
保存手机号
                    save_Deliver(delivers);//保存手机号到订单信息中
                    PrintCC(800,50,"保存成功",HEI,24,1,lightred);
                    delay(500);
                    bar1(800,50,950,100,white);
```

```
    }
    else
    {
         PrintCC(800,50,"长度不合法",HEI,24,1,lightred);
        delay(500);
        bar1(800,50,950,100,white);
    }
}
else if(mouse_press(440, 180, 560, 230)==1)
{
    cur_index = -1;
    press_func(4);//按钮状态切换
    draw_button(1);
    cur_community=1;

}
else if(mouse_press(620, 180, 740, 230)==1)
{
    cur_index = -1;
    press_func(5);//西区
    draw_button(2);
    cur_community=2;
}
else if(mouse_press(800, 180, 920, 230)==1)
{
    cur_index = -1;
    press_func(6);//南区
    draw_button(3);
    cur_community=3;

}
else if(mouse_press(530, 255, 650, 305)==1)
{
```

```
                cur_index = -1;
                press_func(7);//紫菘
                draw_button(4);
                cur_community=4;
            }
            else if(mouse_press(750, 255, 870, 305)==1)
            {
                cur_index = -1;
                press_func(8);//韵苑
                draw_button(5);
                cur_community=5;
            }
            else if (mouse_press(200, 310, 1024, 768) == 1) {

                MouseGet(&mouse);
                mouse_off_arrow(&mouse);
                  returned_index = press_button(mouse.x, mouse.y,
cur_index, cur_community);//获取按钮编号

                 if(returned_index>=0)//如果返回值大于等于 0,则说明选
择了按钮
                {
                                                currentUser.community   =
button[returned_index].commmunity;//获取社区编号
                                                    currentUser.index   =
button[returned_index].index;//获取楼号编号

                       delivers.community=currentUser.community;//保
存社区编号
                        delivers.station=currentUser.index;//保存楼号
编号

                    save_user(currentUser);//保存用户信息
```

```
                    save_Deliver(delivers);//保存订单信息
                }

                cur_index = returned_index;//更新当前按钮编号

                mouse_on_arrow(mouse);

                delay(200);
            }

            if(mouse_press(950, 50, 975,75)==1)
            {
                state = 0;
                mouse_off_arrow(&mouse);
                draw_user_deliver();
                mouse_on_arrow(mouse);
            }
        }
    }
}

void draw_user_deliver(){
    int i,j;
    bar1(200, 0, 1024, 768,white);

    Draw_Rounded_Rectangle(440, 35, 660, 85, 5, 1,deepblue);//取件码输入框
    Draw_Rounded_Rectangle(730, 35, 850, 85, 25, 1,deepblue);//保存按钮

    PrintCC(250,50,"请输入取件码：",HEI,24,1,deepblue);
    PrintCC(250,120,"请选择服务提供商：",HEI,24,1,deepblue);
    PrintCC(250,400,"请选择驿站：",HEI,24,1,deepblue);
```

```
        PrintCC(765,50,"保存",HEI,24,1,deepblue);

        for(i=0;i<3;i++)//绘制服务提供商按钮
        {
            for(j=0;j<3;j++)
            {
                if(i*3+j>=8) break; // 超出快递数量则退出
                        Draw_Rounded_Rectangle(250+250*j,  175+75*i,
250+185+250*j, 175+50+75*i, 5,1,deepblue);
                 PrintText(250+40+250*j,175+13+75*i,companys[i*3+j].na
me,HEI,24,1,deepblue);
            }
        }

        for(i=0;i<3;i++)//绘制驿站地址
        {
            for(j=0;j<3;j++)
            {
                if(i*3+j>=8) break; // 超出驿站数量则退出
                        Draw_Rounded_Rectangle(250+250*j,  455+75*i,
250+185+250*j, 455+50+75*i, 5,1,deepblue);
                 PrintText(250+40+250*j,455+13+75*i,stations[i*3+j].na
me,HEI,24,1,deepblue);
            }
        }

        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5, 1, deepblue);
// 生成订单
        PrintCC(850, 715, "生成订单", HEI, 24, 1, deepblue);
    }

    int choose_company(int x, int y, int* last_index) {
        int i, j;
```

```
        int index = -1;
        int station_count = 8;

        for (i = 0; i < 3; i++) {           // 行
            for (j = 0; j < 3; j++) {       // 列
                int x1 = 250 + 250 * j;
                int y1 = 175 + 75 * i;
                int x2 = x1 + 185;
                int y2 = y1 + 50;

                index = i * 3 + j;
                 if (index >= station_count) break; // 超出快递数量则退
出



                if (x >= x1 && x <= x2 && y >= y1 && y <= y2) {

                    // 恢复上一个按钮
                    if (*last_index != -1 && *last_index != index) {
                        int pre_row = *last_index / 3;
                        int pre_col = *last_index % 3;
                        int pre_x1 = 250 + 250 * pre_col;
                        int pre_y1 = 175 + 75 * pre_row;
                        int pre_x2 = pre_x1 + 185;
                        int pre_y2 = pre_y1 + 50;

                        Fill_Rounded_Rectangle(pre_x1, pre_y1, pre_x2,
pre_y2, 5, white);
                        Draw_Rounded_Rectangle(pre_x1, pre_y1, pre_x2,
pre_y2, 5, 1, deepblue);
                                PrintCC(pre_x1 + 40, pre_y1 + 13,
companys[*last_index].name, HEI, 24, 1, deepblue);
```

```
                    }

                    // 当前按钮高亮
                    Fill_Rounded_Rectangle(x1, y1, x2, y2, 5, deepblue);
                        Draw_Rounded_Rectangle(x1, y1, x2, y2, 5, 1,
deepblue);
                      PrintCC(x1 + 40, y1 + 13, companys[index].name,
HEI, 24, 1, white);

                    *last_index = index;
                    return index + 1; // 返回快递公司编号（1~8）
                }
            }
        }
        return -1; // 未选中任何按钮
    }

    int choose_station(int x, int y, int* last_index_station) {
        int i, j;
        int index = -1;
        int station_count = 8;

        for (i = 0; i < 3; i++) {           // 行
            for (j = 0; j < 3; j++) {       // 列
                int x1 = 250 + 250 * j;
                int y1 = 455 + 75 * i;
                int x2 = x1 + 185;
                int y2 = y1 + 50;

                index = i * 3 + j;
                 if (index >= station_count) break; // 超出快递数量则退
出
```

```
            if (x >= x1 && x <= x2 && y >= y1 && y <= y2) {

                // 恢复上一个按钮
                            if (*last_index_station != -1 &&
*last_index_station != index) {
                    int pre_row = *last_index_station / 3;
                    int pre_col = *last_index_station % 3;
                    int pre_x1 = 250 + 250 * pre_col;
                    int pre_y1 = 455 + 75 * pre_row;
                    int pre_x2 = pre_x1 + 185;
                    int pre_y2 = pre_y1 + 50;

                    Fill_Rounded_Rectangle(pre_x1, pre_y1, pre_x2,
pre_y2, 5, white);
                    Draw_Rounded_Rectangle(pre_x1, pre_y1, pre_x2,
pre_y2, 5, 1, deepblue);
                            PrintText(pre_x1 + 40, pre_y1 + 13,
stations[*last_index_station].name, HEI, 24, 1, deepblue);
                }

                // 当前按钮高亮
                Fill_Rounded_Rectangle(x1, y1, x2, y2, 5, deepblue);
                    Draw_Rounded_Rectangle(x1, y1, x2, y2, 5, 1,
deepblue);
                 PrintText(x1 + 40, y1 + 13, stations[index].name,
HEI, 24, 1, white);

                *last_index_station = index;
                return index + 1; // 返回驿站编号（1~8）
            }
        }
```

```
        }
        return -1; // 未选中任何按钮
    }

    void deliver_input(char *deliver_code,int bar_x1,int bar_y1,int
bar_x2,int bar_y2)
    {
        int length;
        char showtemp[2]= "\0";//存储输入字符,用于输入框展示
         int i=0,k,temp;  // i 为字符个数,temp 为从键盘上读取输入字符的
ACSII 码
        int bDeliver; //光标的横坐标
        int x1,y1;
        x1=bar_x1+4;
        y1=bar_y1+5;//光标相较于输入框的偏移量
        bDeliver=x1+4;//每个字符占 8 个像素,每输入一个字符光标右移 8 个像素

        if(deliver_code[0]=='\0') //如果取件码为空，则显示输入框
            bar1(bar_x1, bar_y1, bar_x2, bar_y2,0xFFFF);
        else
        {            //光标定位至文本末尾
            length=strlen(deliver_code);
            i=length;
            bDeliver+=12*i;
            cursor(bDeliver,y1);
        }

        while(1)
        {
            cursor(bDeliver,y1);
                        if(mouse_location(455,255,845,295)==1    &&
mouse_location(455,335,845,375)==1                               &&
mouse_location(455,415,845,455)==1)
```

```
            mouse_show_cursor(&mouse);
        else
            mouse_show_arrow(&mouse);
        if(bioskey(1)) //如果有键盘输入
        {
            temp=bioskey(0)&0x00ff; //获取键盘输入
              if(temp!='\r'&&temp!='\n')  //检测输入不为回车键，则继续，否则输入结束
            {
                 if((('0'<=temp&&temp<='9')||('a'<=temp&&temp<='z')||('A'<=temp && temp<='Z')||(temp=='-'))&& i <10)//检测为数字或字母或-，则记录
                {
                    hide_cursor(bDeliver,y1); //隐藏原光标

                     deliver_code[i]=temp;//字符送入给定字符串，用于保存取件码信息

                    *showtemp=temp;  //temp 转化为字符串
                     PrintText(bDeliver,y1+2,showtemp,HEI,24,1,0); //显示新的字符串达到画面与实际输入的同步
                    i++;    //字符个数自增

                    deliver_code[i]='\0';//标记字符串结尾

                    bDeliver+=12;   //光标横坐标右移 12 像素
                    draw_cursor(bDeliver,y1);
                }
                else if(temp=='\b'&&i>0)  //检测是否为退格键，是则消除前一个字符
                {
                    hide_cursor(bDeliver,y1);   //隐藏原光标
                    bDeliver-=12;   //光标左移 12 像素
```

```
                    i--;    //字符个数自减

                    deliver_code[i]='\0';//将存储的字符用 0 覆盖

                                bar1(bDeliver,y1,bDeliver+10,  y1+24,
0xffff);   //清空原字符
                    draw_cursor(bDeliver,y1);
                }
                else if(i>=10)
                {
                    mouse_off_arrow(&mouse);
                    mouse_show_arrow(&mouse);
                                PrintCC(750,120," 长 度 超 过 限 制
",HEI,24,1,lightred);
                    delay(500);
                    bar1(750,120,900,160,white);
                }
            }
            else
            {
                break;
            }
        }
        else if (mouse_press(bar_x1,bar_y1,bar_x2,bar_y2)==2)  //
点击框外
        {
            hide_cursor(bDeliver,y1);//隐藏光标
            break;
        }
    }
}

int Deliver_pos(DeliverList DL,Deliver delivers)
```

```
{
    int i=-1;
    for(i=0;i<DL.length;i++)
    {
        if(delivers.id == DL.elem[i].id)
        {
            return i;
        }
    }
    return -1;
}

void DListInsert(DeliverList*DL,Deliver delivers)
{
    Deliver*newbase=NULL;//创建新基址
    if(DL->length>=DL->listsize)//如果线性表已满
    {
        if((newbase=(Deliver*)realloc(DL->elem,(DL->listsize+D_LI
STINCEREMENT)*sizeof(Deliver)))==NULL)////重新分配内存
        {
        CloseSVGA();
        printf("No enough memory!\n");
        printf("DListInsert\n");
        exit(-1);
        }
        DL->elem=newbase;//更新基址
        DL->listsize+=D_LISTINCEREMENT;//更新线性表容量
    }
    *(DL->elem+(DL->length))=delivers;//插入新元素
    DL->length++;//线性表长度加一
}

int save_Deliver(Deliver delivers) {
```

```
int i = 0;
DeliverList DL = {0};
int deliver_pos;
FILE *fp = NULL;

ReadAllDeliver(&DL);//读取所有订单信息

if ((fp = fopen("data\\Deliver.dat", "wb")) == NULL) {
    printf("无法打开文件！\n");
    return -1;
}

// 先查找订单是否已经存在
deliver_pos = Deliver_pos(DL, delivers);

if (deliver_pos == -1)  // 如果订单不存在
{
    DListInsert(&DL, delivers); // 插入订单
}
else  // 如果订单存在，更新原有订单信息
{
    delivers.id = DL.elem[deliver_pos].id; // 保留原订单 ID
    DL.elem[deliver_pos] = delivers;
}

// 重新将线性表写入文件
rewind(fp);//将文件指针移动到文件开头
fwrite(&DL.length, sizeof(short), 1, fp);//写入线性表长度
fwrite(&DL.listsize, sizeof(short), 1, fp);//写入线性表容量

// 逐个写入数据
for (i = 0; i < DL.length; i++) {
    fwrite(&DL.elem[i], sizeof(Deliver), 1, fp);
```

```
    }

    fclose(fp);
    DestroyDList(&DL);
    return 0;
}

void DestroyDList(DeliverList*DL)
{
    free(DL->elem);
    DL->elem=NULL;
    DL->listsize=0;
    DL->length=0;
}

void ReadAllDeliver(DeliverList *DL) {
    int i = 0;
    short length = 0;//线性表初始长度
    short listsize = 10; //线性表初始容量（能够存储 10 个订单）
    FILE *fp = NULL;

    if ((fp = fopen("data\\Deliver.dat", "rb")) == NULL) {//如果打开文件失败
        fp = fopen("data\\Deliver.dat", "wb");// 如果文件不存在则创建一个新的文件
        if (fp == NULL) {//如果创建文件失败
            printf("无法创建文件！\n");
            return;//无法创建文件则返回，不需要继续执行下面的代码
        }
        fwrite(&length, sizeof(short), 1, fp); //如果创建成功则写入初始长度 0
        fwrite(&listsize, sizeof(short), 1, fp); //写入初始容量 10
        fclose(fp);//关闭文件
```

```
            return;//创建完成后返回，不需要继续执行下面的代码
        }
        //如果打开文件成功则读取长度和容量
        fread(&length, sizeof(short), 1, fp);
        fread(&listsize, sizeof(short), 1, fp);
        //把读取的长度和容量赋值给线性表
        DL->length = length;
        DL->listsize = listsize;
        DL->elem = (Deliver *)malloc(listsize * sizeof(Deliver)); //
分配存储空间
        //如果线性表的存储空间分配失败则输出错误信息并退出程序
        if (DL->elem == NULL) {
            printf("No enough memory!\n");
            printf("ReadAllDeliver\n");
            fclose(fp);
            exit(-1);
        }
        //如果分配成功就逐个读取订单数据
        //并把读取的数据存储到线性表中
        for (i = 0; i < length; i++) {
            fread(&DL->elem[i], sizeof(Deliver), 1, fp);
        }

        fclose(fp);
    }
/*****************************************************************/
    #include "all_func.h"

    Food foods[12]={
        {1, 1, "黄焖鸡米饭", 20, "bmp\\canteen\\huang.bmp",0},
        {2, 1, "卤肉饭", 15, "bmp\\canteen\\lurou.bmp",0},
        {3, 1, "蛋炒饭", 10, "bmp\\canteen\\dan.bmp",0},
        {4, 1, "炒河粉", 10, "bmp\\canteen\\hefen.bmp",0},
```

```
    {5, 1, "牛肉拉面", 12, "bmp\\canteen\\niurou.bmp",0},
    {6, 1, "番茄鸡蛋面", 8, "bmp\\canteen\\fanqie.bmp",0},
    {7, 1, "红烧肉", 12, "bmp\\canteen\\hong.bmp",0},
    {8, 1, "炒油麦菜", 4, "bmp\\canteen\\cai.bmp",0},
    {9, 1, "小米粥", 3, "bmp\\canteen\\xiaomi.bmp",0},
    {10, 1, "八宝粥", 3, "bmp\\canteen\\babao.bmp",0},
    {11, 1, "排骨炖藕汤", 8, "bmp\\canteen\\ou.bmp",0},
    {12, 1, "紫菜蛋花汤", 2, "bmp\\canteen\\zicai.bmp",0}
};
FoodCart food_carts[12]={0};//购物车内的商品
ShoppingFood shopping_food={0};//购物车整体

void user_food(int index){
    int foodCount=12;
    int state=0;//0 为未选择排序，1 为选择排序

    mouse_off_arrow(&mouse);

    draw_user_food(index);

    mouse_on_arrow(mouse);

    while(1){
        mouse_show_arrow(&mouse);

        if(mouse_press(40, 113, 160, 163)==1)
        {
            int i;
            for(i=0;i<foodCount;i++)
            {
                foods[i].quantity=0;//清空商品数量
            }
            shopping_food.itemCount=0;//清空购物车
```

```
            shopping_food.items=NULL;//清空购物车

            return;
        }
        else if(mouse_press(800, 700, 1000, 750)==1)//查看订单
        {
            food_order(index);//查看订单

            //return 后从这开始
            mouse_off_arrow(&mouse);
            bar1(200, 0, 1024, 768, white); // 清除注册界面残留
            draw_user_food(index);
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(270, 235, 1070, 835)==1)//点击商品
        {
            MouseGet(&mouse);
                AddSub_food(mouse.x, mouse.y, foodCount, foods,
food_carts, &shopping_food.itemCount);
            draw_food_quantity(foods); //刷新页面显示更新后的数量
            delay(100);
        }
        else if(mouse_press(220,75, 250, 90)==1)
        {
            mouse_off_arrow(&mouse);//隐藏鼠标
            draw_sort();//绘制排序页面
            mouse_on_arrow(mouse);//显示鼠标
            state=1;//已点击排序
        }

        if (state == 1) {
            if (mouse_press(205, 95, 445, 144) == 1) // 点击从高到
低
```

```
{
    int i, j;

    for (i = 0; i < 12 - 1; i++) {
        for (j = 0; j < 12 - 1 - i ; j++) {
            if (foods[j].price < foods[j + 1].price) {
                Food temp = foods[j];
                foods[j] = foods[j + 1];
                foods[j + 1] = temp;
            }
        }
    }

    mouse_off_arrow(&mouse);//隐藏鼠标
    draw_user_food(index);
    mouse_on_arrow(mouse);//显示鼠标
    state=0;
}
 else if(mouse_press(205, 146, 445, 295)==1)//点击从低
到高
{
    int i, j;

    for (i = 0; i < 12 - 1; i++) {
        for (j = 0; j < 12 - 1 - i ; j++) {
            if (foods[j].price > foods[j + 1].price) {
                Food temp = foods[j];
                foods[j] = foods[j + 1];
                foods[j + 1] = temp;
            }
        }
    }
```

```
                    mouse_off_arrow(&mouse);//隐藏鼠标
                    draw_user_food(index);
                    mouse_on_arrow(mouse);//显示鼠标
                    state=0;
                }

            }
        }
    }

    void draw_user_food(int index){
        int cnt=0;
        int i,j;
        bar1(200, 0, 1024, 768, white);
        bar1(0, 250, 199, 768, deepblue);

        Line_Thick(220,75,235,90,1,black);//
        Line_Thick(235,90,250,75,1,black);//

        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5,1,deepblue);//查看订单按钮

        PrintCC(850,715,"查看订单",HEI,24,1,deepblue);

            PrintCC(220,700,  canteen[index-1].name,  HEI,  48,  1,deepblue);//显示食堂名称

        //显示商品信息
        for(j=0;j<3;j++){
            for(i=0;i<4;i++){//先横向，再竖向

                char quantity_str[20];
                char price_str[20];
```

```
                // 使用 sprintf 将 quantity 转换为字符串
                sprintf(quantity_str, "%d", foods[cnt].quantity);
                sprintf(price_str, "%.2f", foods[cnt].price);

                Line_Thick(270+200*i, 235+200*j, 290+200*i, 235+200*j,
1, deepblue);//减号

                Line_Thick(390+200*i, 235+200*j, 370+200*i, 235+200*j,
1, deepblue);//加号
                Line_Thick(380+200*i, 225+200*j, 380+200*i, 245+200*j,
1, deepblue);


                 PrintCC(270+200*i,75+200*j,foods[cnt].name,HEI,24,1,b
lack);//显示商品名称

                  PrintText(270+22+200*i, 220+3+200*j,price_str, HEI,
24, 0, black);//显示商品价格
                              PrintText(395+200*i,75+200*j,   (unsigned
char*)quantity_str,HEI,24,0,black);//显示商品数量


                Readbmp64k(270+200*i, 100+200*j, foods[cnt].photo);//
显示商品图片
                cnt++;

            }
        }

    }

    //重新显示商品数量
    void draw_food_quantity(Food foods[]){
```

```
        int i=0;
        int j=0;
        int cnt=0;
        //显示商品信息
        for(j=0;j<3;j++){
            for(i=0;i<4;i++){//先横向，再竖向
                char quantity_str[20];
                // 使用 sprintf 将 quantity 转换为字符串
                sprintf(quantity_str, "%d", foods[cnt].quantity);
                 // 调用 PrintText 函数，将 quantity_str 转换为 unsigned
char* 类型
                bar1(395+200*i, 75+200*j, 460+200*i, 95+200*j,white);
                            PrintText(395+200*i,75+200*j,   (unsigned
char*)quantity_str,HEI,24,0,black);//显示商品数量
                cnt++;
            }
        }
    }

    //加减商品
    void AddSub_food(int mx, int my, int foodCount, Food foods[],
FoodCart food_carts[], int* itemCount) {
        int i, j, index;
        int baseX = 270, baseY = 235;
        int width = 200, height = 200;

        for (j = 0; j < 3; j++) {
            for (i = 0; i < 4; i++) {
                index = i + j * 4;
                if (index >= foodCount) return;// 超出商品数量，退出

                // 加号区域
                if (mx >= 370 + i * width && mx <= 390 + i * width &&
```

```
                    my >= 225 + j * height && my <= 245 + j * height)
{
                            addToCart_food(foods[index],  food_carts,
itemCount,index);
                    foods[index].quantity++;
                    return;
                }
                // 减号区域
                if (mx >= 270 + i * width && mx <= 290 + i * width &&
                    my >= 225 + j * height && my <= 245 + j * height)
{
                    if (foods[index].quantity > 0) {
                        foods[index].quantity--;  // 商品页面数量 -1
                         removeFromCart_food(foods[index], food_carts,
itemCount);
                    }else {
                                PrintCC(220,660,"此商品已从购物车中移除
",HEI,24,1,lightred);
                        delay(500);
                        bar1(220,660,800,700,white);
                    }
                    return;
                }
            }
        }
    }

    void   addToCart_food(Food   f,   FoodCart   food_carts[],   int
*itemCount,int index) {
        int i=0;
        for (i = 0; i < *itemCount; i++) {
            if (food_carts[i].id == f.id) {
                food_carts[i].quantity++;
```

```
                return;
            }
        }
        food_carts[*itemCount].id = f.id;
        strcpy(food_carts[*itemCount].name, f.name);
        strcpy(food_carts[*itemCount].photo, f.photo);
        food_carts[*itemCount].price=f.price;
        food_carts[*itemCount].quantity = 1;
         food_carts[*itemCount].index_in_foods=index;//在商品数组中的索
引
        (*itemCount)++;
    }

    void  removeFromCart_food(Food  f,  FoodCart  food_carts[],  int
*itemCount) {
        int i, j;
        for (i = 0; i < *itemCount; i++) {
            if (food_carts[i].id == f.id) {
                if (food_carts[i].quantity > 1) {
                    food_carts[i].quantity--;
                } else if (food_carts[i].quantity == 1) {
                    // 如果减到 0，删除这个商品
                    for (j = i; j < *itemCount - 1; j++) {
                        food_carts[j] = food_carts[j + 1];
                    }
                    (*itemCount)--;
                }
                return;
            }
        }
    }
/*****************************************************************/
    #include "all_func.h"
```

```
    Order orders = {0}; // 订单

    void user_order(){

        UserList UL = {0};
        USER currentUser;

        int page = 0;// 初始页码
        int totalPage =(cart.itemCount - 6 + 11 ) / 12 + 1 ; // 总页数
(向上取整)
        int state = 0; // 判断是否需要完善信息

        int cur_index = -1;
        int cur_community=0;
        int returned_index;

        ReadAllUser(&UL); // 读取用户列表

        currentUser=UL.elem[users.pos];// 获取当前用户信息

        DestroyUList(&UL); // 释放用户列表空间

        mouse_off_arrow(&mouse);

        draw_user_order(page);

        mouse_on_arrow(mouse);

        while(1){
            mouse_show_arrow(&mouse);

            if (mouse_press(40, 113, 160, 163) == 1)
```

```
            {
                //user_cart();// 返回用户购物车页面
                return;
            }
            if(state==0)
            {
                if (mouse_press(220, 700, 340, 750) == 1)
                {
                    if (page > 0) {
                        page--;
                        draw_user_order(page);// 绘制用户订单页面
                    } else {
                        // 提示：已是第一页
                           PrintCC(550, 25, "已是第一页", HEI, 24, 1,
lightred);
                        delay(500);
                        bar1(550, 25, 700, 60, white);
                    }
                }
                else if (mouse_press(420, 700, 540, 750) == 1)
                {
                    if (page < totalPage - 1) {
                        page++;
                        draw_user_order(page);// 绘制用户订单页面
                    } else {
                        // 提示：已是最后一页
                          PrintCC(550, 25, "已是最后一页", HEI, 24, 1,
lightred);
                        delay(500);
                        bar1(550, 25, 700, 60, white);
                    }
                }
                else if(mouse_press(800, 700, 1000, 750) == 1)
```

```
                {
                            if (currentUser.community == '\0' ||
strlen(currentUser.number) == 0)// 判断用户信息是否完善
                    {
                        mouse_off_arrow(&mouse);
                        draw_info();
                        mouse_on_arrow(mouse);
                        state = 1;
                    }
                    else
                    {
                        save_order(orders); // 保存订单
                           PrintCC(800, 50, "订单已保存", HEI, 24, 1,
lightred);
                        delay(500);
                        bar1(800, 50, 1024, 100, white);
                    }
                }
            }
            // 完善用户信息
            if(state==1)
            {
                if(mouse_press(430, 105, 650, 155)==1)
                {
                      number_input(currentUser.number, 435, 110, 645,
150); // 输入手机号
                }
                else if(mouse_press(710, 105, 830, 155)==1)
                {
                    if(strlen(currentUser.number)==11)
                    {
                        save_user(currentUser);
                        PrintCC(800,50,"保存成功",HEI,24,1,lightred);
```

```
            delay(500);
            bar1(800,50,950,100,white);
        }
        else
        {
             PrintCC(800,50,"长度不合法",HEI,24,1,lightred);
            delay(500);
            bar1(800,50,950,100,white);
        }
    }
    else if(mouse_press(440, 180, 560, 230)==1)
    {
        cur_index = -1;
        press_func(4);//按钮状态切换
        draw_button(1);
        cur_community=1;

    }
    else if(mouse_press(620, 180, 740, 230)==1)
    {
        cur_index = -1;
        press_func(5);//西区
        draw_button(2);
        cur_community=2;
    }
    else if(mouse_press(800, 180, 920, 230)==1)
    {
        cur_index = -1;
        press_func(6);//南区
        draw_button(3);
        cur_community=3;

    }
```

```
                else if(mouse_press(530, 255, 650, 305)==1)
                {
                    cur_index = -1;
                    press_func(7);//紫菘
                    draw_button(4);
                    cur_community=4;
                }
                else if(mouse_press(750, 255, 870, 305)==1)
                {
                    cur_index = -1;
                    press_func(8);//韵苑
                    draw_button(5);
                    cur_community=5;
                }
                else if (mouse_press(200, 310, 1024, 768) == 1) {
                    MouseGet(&mouse);
                    mouse_off_arrow(&mouse);

                      returned_index = press_button(mouse.x, mouse.y,
cur_index, cur_community);//获取按钮编号

                     if(returned_index>=0)//如果返回值大于等于 0,则说明选
择了按钮
                    {
                                                currentUser.community   =
button[returned_index].commmunity;//获取社区编号//必需，不能删

                                                    currentUser.index   =
button[returned_index].index;//获取楼号编号

                        save_user(currentUser);//保存用户信息
                    }
```

```
                cur_index = returned_index;//更新当前按钮编号

                mouse_on_arrow(mouse);

                delay(200);
            }

            if(mouse_press(950, 50, 975,75)==1)
            {
                state = 0;
                mouse_off_arrow(&mouse);
                draw_user_order(page);
                mouse_on_arrow(mouse);
            }
        }
    }
}

void draw_user_order(int page){
    int i;
    UserList UL = {0};
    OrderList OL = {0};
    USER currentUser;

    char* current_time = get_current_time(); // 获取当前时间
    char time_str[100]; // 打印下单时间
    char user_name[100]; // 打印用户名
    char user_phone[100]; // 打印用户手机号
    char address[100]; // 打印用户地址
    int startIdx = 0;// 起始商品索引
    int itemsPerPage = 0;// 每页商品数量
    int endIdx = 0;// 结束商品索引
    int item_y = 0;// 商品框的 y 坐标
```

```
        float total_amount = 0.0; // 总金额
        char total_str[50]; // 总金额字符串
        int fullPageItemCount = 0; // 满页商品数量

        ReadAllUser(&UL); // 读取用户列表
        currentUser = UL.elem[users.pos]; // 获取当前用户信息

        ReadAllOrder(&OL); // 读取订单列表
        orders.id = OL.length + 1; // 订单号

        sprintf(time_str, "下单时间：%s", current_time);
        sprintf(user_name, "用户名：%s", currentUser.name);
        sprintf(user_phone, "手机号：%s", currentUser.number);

        bar1(200, 0, 1024, 768, white); // 清空屏幕

        // 分页按钮
        Draw_Rounded_Rectangle(220, 700, 340, 750, 25, 1, deepblue);
// 上一页
        Draw_Rounded_Rectangle(420, 700, 540, 750, 25, 1, deepblue);
// 下一页
        PrintCC(245, 715, "上一页", HEI, 24, 1, deepblue);
        PrintCC(445, 715, "下一页", HEI, 24, 1, deepblue);

        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5, 1, deepblue);
// 确认并支付
        PrintCC(830, 715, "确认并支付", HEI, 24, 1, deepblue);

        // 页头信息只在第一页显示
        if (page == 0) {
            char order_number_str[20]; // 订单号字符串
            char community[50]; // 社区字符串
```

```
            char building[50]; // 楼栋字符串
            sprintf(order_number_str, "订单号：%d", orders.id); // 订单
号
            PrintText(250, 50, order_number_str, HEI, 24, 1, black);
            PrintText(250, 100, time_str, HEI, 24, 1, black);
            PrintText(250, 150, user_name, HEI, 24, 1, black);
            PrintText(250, 200, user_phone, HEI, 24, 1, black);

            if (currentUser.index == 0)
                sprintf(address,"地址：未绑定地址");
            else
                                sprintf(address,   "  地  址  :  %s",
node[currentUser.index].name); // 用户地址

            PrintText(250, 250, address, HEI, 24, 1, black);

            // 表头
            PrintCC(250, 300, "商品详情：", HEI, 24, 1, black);
            PrintCC(750, 300, "数量：", HEI, 24, 1, black);
            PrintCC(900, 300, "金额：", HEI, 24, 1, black);
            PrintText(250, 320, "------------------------------", HEI,
32, 1, black);// 分隔线

            startIdx = 0;
            itemsPerPage = 6;
        } else {
            startIdx = 6 + (page - 1) * 12;
            itemsPerPage = 12;
        }

        endIdx = startIdx + itemsPerPage;
        if (endIdx > cart.itemCount)// 防止越界
            endIdx = cart.itemCount;
```

```
        item_y = (page == 0) ? 350 : 50;
        for (i = startIdx; i < endIdx; i++) {
            char total_str[50]; // 商品总价
            char quantity_str[20]; // 商品数量
            int productIndex = carts[i].index_in_products; // 商品索引
            int quantity = products[productIndex].quantity;

            sprintf(total_str, "%.2f", products[productIndex].price *
quantity);
            sprintf(quantity_str, "x%d", quantity);

            PrintCC(250, item_y, carts[i].name, HEI, 24, 1, black); //
商品名
            PrintText(750, item_y, (unsigned char*)quantity_str, HEI,
24, 1, black);// 商品数量
            PrintText(900, item_y, (unsigned char*)total_str, HEI, 24,
1, black);// 商品金额

            item_y += 50;
        }

        // 判断是否需要在此页显示总金额（当前页没有满）
        fullPageItemCount = (page == 0) ? 6 : 12;// 第一页显示 6 个商品，
其余页显示 12 个商品
                    if       ((endIdx       -       startIdx)       <
fullPageItemCount||endIdx==cart.itemCount) {// 当前页商品数量不满一页或
最后一个商品刚好满页都要打印出总金额
            //如果不是最后一个商品但是满页就不打印总金额
            // 打印分隔线
            PrintText(250, item_y - 30, "----------------------------
---", HEI, 32, 1, black);
```

```
            // 计算总金额
            total_amount = 0.0;
            for (i = 0; i < cart.itemCount; i++) {
                int productIndex = carts[i].index_in_products;
                int quantity = products[productIndex].quantity;
                     total_amount  +=  products[productIndex].price  *
quantity;
                carts[i].quantity = quantity; // 记录购物车商品数量
                carts[i].price = products[productIndex].price; // 记录
商品价格
            }

            sprintf(total_str, "总金额：%.2f 元", total_amount);
            PrintText(750, item_y + 10, total_str, HEI, 24, 1, black);
        }

        //存储订单信息

        strcpy(orders.order_time, current_time); // 下单时间
        strcpy(orders.user_name, currentUser.name); // 用户名
        strcpy(orders.user_phone, currentUser.number); // 用户手机号
        // orders.destination = currentUser.community; // 用户地址
        orders.community=currentUser.community; // 用户社区
        orders.destination=currentUser.index;

        if ( orders.destination >= 21 && orders.destination <=55 )
        orders.pick_up_location = 20;
        else if ( orders.destination >=56 && orders.destination <= 71 )
        orders.pick_up_location = 19;
        else if ( orders.destination >=72 && orders.destination <= 98 )
        orders.pick_up_location = 18;
         else  if  (  orders.destination  >=99  &&  orders.destination  <=
115 )
```

```
        for (i = 0; i < cart.itemCount; i++) {
            orders.item[i] = carts[i]; // 购物车内商品信息
        }
        orders.itemCount = cart.itemCount; // 购物车内商品数量
        orders.total_amount = total_amount; // 总金额

        DestroyUList(&UL); // 释放用户列表空间
        DestroyOList(&OL); // 释放订单列表空间

    }

    void draw_info(){

        bar1(200, 0, 1024, 768, white);//清屏
        Draw_Rounded_Rectangle(225, 25, 1000, 750, 30, 2,0x6B4D);//最外围灰色圆角矩形框

        Line_Thick(950, 50, 975, 75, 1, black);//
        Line_Thick(950, 75, 975, 50, 1, black);//

        PrintCC(250, 50, "请先完善个人信息", HEI, 24, 1, lightred);

        Draw_Rounded_Rectangle(440, 180, 560, 230, 25, 1,deepblue);//东区按钮
        Draw_Rounded_Rectangle(620, 180, 740, 230, 25, 1,deepblue);//西区按钮
        Draw_Rounded_Rectangle(800, 180, 920, 230, 25, 1,deepblue);//南区按钮
        Draw_Rounded_Rectangle(530, 255, 650, 305, 25, 1,deepblue);//紫菘按钮
        Draw_Rounded_Rectangle(750, 255, 870, 305, 25, 1,deepblue);//韵苑按钮
```

```
        Draw_Rounded_Rectangle(430, 105, 650, 155, 5, 1,deepblue);//
手机号输入框
        Draw_Rounded_Rectangle(710, 105, 830, 155, 25, 1,deepblue);//
保存按钮

        PrintCC(250,120,"请输入手机号：",HEI,24,1,deepblue);
        PrintCC(250,190,"请选择住址：",HEI,24,1,deepblue);
        PrintCC(745,120,"保存",HEI,24,1,deepblue);
        PrintCC(475,195,"东区",HEI,24,1,deepblue);
        PrintCC(655,195,"西区",HEI,24,1,deepblue);
        PrintCC(835,195,"南区",HEI,24,1,deepblue);
        PrintCC(565,270,"紫菘",HEI,24,1,deepblue);
        PrintCC(785,270,"韵苑",HEI,24,1,deepblue);
        PrintCC(745,120,"保存",HEI,24,1,deepblue);

    }
    // 获取当前时间并转换为字符串
    char* get_current_time() {
        time_t rawtime;
        struct tm * timeinfo;
        static char buffer[20];

        // 获取当前时间
        time(&rawtime);
        // 将时间转换为本地时间
        timeinfo = localtime(&rawtime);

        // 将时间转换为字符串
           strftime(buffer,   sizeof(buffer),   "%Y-%m-%d   %H:%M:%S",
timeinfo);
        return buffer;
    }
```

```
/*****************************
功能说明：得到元素在线性表中的位置
参数说明：线性表，元素
返回值：  如果存在就返回位置，否则返回-1
*****************************/

int Order_pos(OrderList OL,Order orders)
{
    int i=-1;
    for(i=0;i<OL.length;i++)
    {
        if(orders.id == OL.elem[i].id)
        {
            return i;
        }
    }
    return -1;
}

/*****************************
功能说明：在线性表 L 末尾插入元素
参数说明：线性表地址，要插入的元素
返回值：  无
*****************************/
void OListInsert(OrderList*OL,Order orders)
{
    Order*newbase=NULL;//创建新基址
    if(OL->length>=OL->listsize)//如果线性表已满
    {
        if((newbase=(Order*)realloc(OL->elem,(OL->listsize+O_LIST
INCEREMENT)*sizeof(Order)))==NULL)////重新分配内存
        {
```

```
        CloseSVGA();
        printf("No enough memory!\n");
        printf("OListInsert\n");
        exit(-1);
        }
        OL->elem=newbase;//更新基址
        OL->listsize+=O_LISTINCEREMENT;//更新线性表容量
    }
    *(OL->elem+(OL->length))=orders;//插入新元素
    OL->length++;//线性表长度加一
}

/**********************
功能说明：保存账单信息函数
参数说明：账单结构体
返回值说明:0：保存成功   -1：  保存失败
**********************/
int save_order(Order orders) {
    int i = 0;
    OrderList OL = {0};
    int order_pos;
    FILE *fp = NULL;

    ReadAllOrder(&OL);//读取所有订单信息

    if ((fp = fopen("data\\order.dat", "wb")) == NULL) {
        printf("无法打开文件！\n");
        return -1;
    }

    // 先查找订单是否已经存在
    order_pos = Order_pos(OL, orders);
```

```
    if (order_pos == -1)  // 如果订单不存在
    {
        OListInsert(&OL, orders); // 插入订单
    }
    else  // 如果订单存在，更新原有订单信息
    {
        orders.id = OL.elem[order_pos].id; // 保留原订单 ID
        OL.elem[order_pos] = orders;
    }

    // 重新将线性表写入文件
    rewind(fp);//将文件指针移动到文件开头
    fwrite(&OL.length, sizeof(short), 1, fp);//写入线性表长度
    fwrite(&OL.listsize, sizeof(short), 1, fp);//写入线性表容量

    // 逐个写入数据
    for (i = 0; i < OL.length; i++) {
        fwrite(&OL.elem[i], sizeof(Order), 1, fp);
    }

    fclose(fp);
    DestroyOList(&OL);
    return 0;
}

void DestroyOList(OrderList*OL)
{
    free(OL->elem);
    OL->elem=NULL;
    OL->listsize=0;
    OL->length=0;
}
```

```
// 初始化线性表
void ReadAllOrder(OrderList *OL) {
    int i = 0;
    short length = 0;//线性表初始长度
    short listsize = 10; //线性表初始容量（能够存储 10 个订单）
    FILE *fp = NULL;

    if ((fp = fopen("data\\order.dat", "rb")) == NULL) {//如果打开文件失败
        fp = fopen("data\\order.dat", "wb");// 如果文件不存在则创建一个新的文件
        if (fp == NULL) {//如果创建文件失败
            printf("无法创建文件！\n");
            return;//无法创建文件则返回，不需要继续执行下面的代码
        }
        fwrite(&length, sizeof(short), 1, fp); //如果创建成功则写入初始长度 0
        fwrite(&listsize, sizeof(short), 1, fp); //写入初始容量 10
        fclose(fp);//关闭文件
        return;//创建完成后返回，不需要继续执行下面的代码
    }
    //如果打开文件成功则读取长度和容量
    fread(&length, sizeof(short), 1, fp);
    fread(&listsize, sizeof(short), 1, fp);
    //把读取的长度和容量赋值给线性表
    OL->length = length;
    OL->listsize = listsize;
    OL->elem = (Order *)malloc(listsize * sizeof(Order)); //分配存储空间
    //如果线性表的存储空间分配失败则输出错误信息并退出程序
    if (OL->elem == NULL) {
        printf("No enough memory!\n");
        printf("ReadAllOrder\n");
```

```
            fclose(fp);
            exit(-1);
        }
        //如果分配成功就逐个读取订单数据
        //并把读取的数据存储到线性表中
        for (i = 0; i < length; i++) {
            fread(&OL->elem[i], sizeof(Order), 1, fp);
        }

        fclose(fp);
    }
/*****************************************************************/
    #include "all_func.h"

    Product products[84]=
    {
    /*1=============================================================
=============*/
        {1, 1, "盆", 5.00, "bmp\\shop\\pen.bmp", 0},
        {2, 1, "扫把", 12.00, "bmp\\shop\\saoba.bmp", 0},
        {3, 1, "餐具", 10.00, "bmp\\shop\\canju.bmp", 0},
        {4, 1, "碗", 3.00, "bmp\\shop\\wan.bmp", 0},
        {5, 1, "水杯", 15.00, "bmp\\shop\\shuibei.bmp", 0},
        {6, 1, "衣架", 5.00, "bmp\\shop\\yijia.bmp", 0},
        {7, 1, "牙刷", 10.00, "bmp\\shop\\yashua.bmp", 0},
        {8, 1, "拖把", 18.00, "bmp\\shop\\tuoba.bmp", 0},
        {9, 1, "枕头", 20.00, "bmp\\shop\\zhentou.bmp", 0},
        {10, 1, "毛巾", 12.00, "bmp\\shop\\maojin.bmp", 0},
        {11, 1, "挂钩", 1.00, "bmp\\shop\\guagou.bmp", 0},
        {12, 1, "马桶塞子", 10.00, "bmp\\shop\\matong.bmp", 0},
    /*2=============================================================
=============*/
        {13, 2, "黑笔", 2.00, "bmp\\shop\\blackbi.bmp", 0},
```

```
        {14, 2, "红笔", 3.00, "bmp\\shop\\redbi.bmp", 0},
        {15, 2, "铅笔", 4.00, "bmp\\shop\\qianbi.bmp", 0},
        {16, 2, "钢笔", 35.00, "bmp\\shop\\gangbi.bmp", 0},
        {17, 2, "剪刀", 12.00, "bmp\\shop\\jiandao.bmp", 0},
        {18, 2, "橡皮", 2.00, "bmp\\shop\\xiangpi.bmp", 0},
        {19, 2, "尺子", 8.00, "bmp\\shop\\chizi.bmp", 0},
        {20, 2, "胶带", 2.00, "bmp\\shop\\jiaodai.bmp", 0},
        {21, 2, "固体胶", 4.00, "bmp\\shop\\jiao.bmp", 0},
        {22, 2, "修正带", 6.00, "bmp\\shop\\xiuzheng.bmp", 0},
        {23, 2, "笔记本", 3.00, "bmp\\shop\\benzi.bmp", 0},
        {24, 2, "订书机", 15.00, "bmp\\shop\\dingshu.bmp", 0},
    /*3============================================================
=============*/
        {25, 3, "薯片", 7.00, "bmp\\shop\\shupian.bmp", 0},
        {26, 3, "达利园蛋糕", 1.00, "bmp\\shop\\dali.bmp", 0},
        {27, 3, "奥利奥饼干", 8.00, "bmp\\shop\\aoliao.bmp", 0},
        {28, 3, "辣条", 5.00, "bmp\\shop\\latiao.bmp", 0},
        {29, 3, "大白兔奶糖", 1.00, "bmp\\shop\\dabaitu.bmp", 0},
        {30, 3, "卤鹌鹑蛋", 1.00, "bmp\\shop\\ludan.bmp", 0},
        {31, 3, "巧克力", 1.00, "bmp\\shop\\defu.bmp", 0},
        {32, 3, "锅巴", 1.00, "bmp\\shop\\guoba.bmp", 0},
        {33, 3, "吐司面包", 1.00, "bmp\\shop\\tusi.bmp", 0},
        {34, 3, "鸡腿", 1.00, "bmp\\shop\\jitui.bmp", 0},
        {35, 3, "方便面", 1.00, "bmp\\shop\\paomian.bmp", 0},
        {36, 3, "薄荷糖", 1.00, "bmp\\shop\\bohe.bmp", 0},
    /*4============================================================
=============*/
        {37, 4, "可口可乐", 3.00, "bmp\\shop\\kekou.bmp", 0},
        {38, 4, "百事可乐", 3.00, "bmp\\shop\\baishi.bmp", 0},
        {39, 4, "芬达", 3.00, "bmp\\shop\\fenda.bmp", 0},
        {40, 4, "阿萨姆奶茶", 4.00, "bmp\\shop\\asamu.bmp", 0},
        {41, 4, "茶兀", 5.00, "bmp\\shop\\chapi.bmp", 0},
        {42, 4, "脉动", 5.00, "bmp\\shop\\maidong.bmp", 0},
```

```
        {43, 4, "雪碧", 3.00, "bmp\\shop\\xuebi.bmp", 0},
        {44, 4, "冰红茶", 3.00, "bmp\\shop\\bing.bmp", 0},
        {45, 4, "绿茶", 3.00, "bmp\\shop\\lv.bmp", 0},
        {46, 4, "优酸乳", 2.00, "bmp\\shop\\you.bmp", 0},
        {47, 4, "纯牛奶", 4.00, "bmp\\shop\\milk.bmp", 0},
        {48, 4, "酸奶", 5.00, "bmp\\shop\\suan.bmp", 0},
    /*5============================================================
=============*/
        {49, 5, "篮球", 10.00, "bmp\\shop\\lanqiu.bmp", 0},
        {50, 5, "足球", 10.00, "bmp\\shop\\zuqiu.bmp", 0},
        {51, 5, "羽毛球", 10.00, "bmp\\shop\\yu.bmp", 0},
        {52, 5, "乒乓球", 10.00, "bmp\\shop\\ping.bmp", 0},
        {53, 5, "网球", 10.00, "bmp\\shop\\wang.bmp", 0},
        {54, 5, "排球", 10.00, "bmp\\shop\\pai.bmp", 0},
        {55, 5, "网球拍", 10.00, "bmp\\shop\\wangpai.bmp", 0},
        {56, 5, "羽毛球拍", 10.00, "bmp\\shop\\yupai.bmp", 0},
        {57, 5, "乒乓球拍", 10.00, "bmp\\shop\\pingpai.bmp", 0},
        {58, 5, "跳绳", 10.00, "bmp\\shop\\tiao.bmp", 0},
        {59, 5, "哑铃", 10.00, "bmp\\shop\\ya.bmp", 0},
        {60, 5, "运动背包", 50.00, "bmp\\shop\\beibao.bmp", 0},
    /*6============================================================
=============*/
        {61, 6, "苹果", 10.00, "bmp\\shop\\apple.bmp", 0},
        {62, 6, "哈密瓜", 10.00, "bmp\\shop\\hami.bmp", 0},
        {63, 6, "梨", 10.00, "bmp\\shop\\pear.bmp", 0},
        {64, 6, "橘子", 10.00, "bmp\\shop\\orange.bmp", 0},
        {65, 6, "草莓", 10.00, "bmp\\shop\\cao.bmp", 0},
        {66, 6, "西瓜", 10.00, "bmp\\shop\\xigua.bmp", 0},
        {67, 6, "火龙果", 10.00, "bmp\\shop\\huo.bmp", 0},
        {68, 6, "芒果", 10.00, "bmp\\shop\\mango.bmp", 0},
        {69, 6, "猕猴桃", 10.00, "bmp\\shop\\mihou.bmp", 0},
        {70, 6, "蓝莓", 10.00, "bmp\\shop\\lan.bmp", 0},
        {71, 6, "樱桃", 10.00, "bmp\\shop\\yingtao.bmp", 0},
```

```
        {72, 6, "圣女果", 10.00, "bmp\\shop\\shengnv.bmp", 0},
    /*7=============================================================
=============*/
        {73, 7, "笔记本套装", 10.00, "bmp\\shop\\bijitao.bmp", 0},
        {74, 7, "茶杯套装", 10.00, "bmp\\shop\\chatao.bmp", 0},
        {75, 7, "书签套装", 10.00, "bmp\\shop\\shutao.bmp", 0},
        {76, 7, "帆布包", 10.00, "bmp\\shop\\fanbu.bmp", 0},
        {77, 7, "钥匙扣", 10.00, "bmp\\shop\\yaoshi.bmp", 0},
        {78, 7, "明信片", 10.00, "bmp\\shop\\mingxin.bmp", 0},
        {79, 7, "文创直尺", 10.00, "bmp\\shop\\wenchi.bmp", 0},
        {80, 7, "冰箱贴", 10.00, "bmp\\shop\\bingx.bmp", 0},
        {81, 7, "文化衫", 10.00, "bmp\\shop\\wenshan.bmp", 0},
        {82, 7, "金属书签", 10.00, "bmp\\shop\\jinqian.bmp", 0},
        {83, 7, "金属吊坠", 10.00, "bmp\\shop\\jinzhui.bmp", 0},
        {84, 7, "校徽", 10.00, "bmp\\shop\\xiaohui.bmp", 0},
    /*=============================================================
============*/
    };
    CartItem carts[84]={0};
    ShoppingCart cart={0};

    void user_shop(){
        int productCount = 84;//超市里商品数量初始化
        int currentpage = 1;//当前页面初始化
        int state=0;//0 为未点击排序，1 为已点击排序

        mouse_off_arrow(&mouse);//隐藏鼠标

         draw_user_shop(products, productCount,currentpage);//绘制用户
超市页面

        mouse_on_arrow(mouse);//显示鼠标
```

```
while(1){
    mouse_show_arrow(&mouse);

    if(mouse_press(40, 113, 160, 163)==1)
    {
        int i;
        for(i=0;i<productCount;i++)
        {
            products[i].quantity=0;//清空商品数量
        }
        cart.itemCount=0;//清空购物车
        cart.items=NULL;//清空购物车

        return;
        //welcome();//首页
    }
    else if(mouse_press(40, 276, 160, 326)==1)
    {
        press_func(1);//进入超市页面
        user_shop();//用户超市页面
        return;
    }
    else if(mouse_press(40, 439, 160, 489)==1)
    {
        press_func(2);//进入外卖页面
        user_takeout();//用户外卖页面
        return;
    }
    else if(mouse_press(40, 602, 160, 652)==1)
    {
        press_func(3);//进入快递页面
        user_deliver();//用户快递页面
        return;
```

```
            }
            else if(mouse_press(800, 700, 1000, 750)==1)
            {
                user_cart();//用户购物车页面
                break;
            }else if(mouse_press(220,75, 250, 90)==1)
            {
                mouse_off_arrow(&mouse);//隐藏鼠标
                draw_sort();//绘制排序页面
                mouse_on_arrow(mouse);//显示鼠标
                state=1;//已点击排序
            }
            else if(mouse_press(200, 0, 320, 50)==1)
            {
                currentpage = 1;//生活用品
                mouse_off_arrow(&mouse);//隐藏鼠标
                 draw_user_shop(products, productCount,currentpage);//
绘制用户超市页面
                mouse_on_arrow(mouse);//显示鼠标

            }
            else if(mouse_press(320, 0, 440, 50)==1)
            {
                currentpage = 2;//文具
                mouse_off_arrow(&mouse);//隐藏鼠标
                 draw_user_shop(products, productCount,currentpage);//
绘制用户超市页面
                mouse_on_arrow(mouse);//显示鼠标
            }
            else if(mouse_press(440, 0, 560, 50)==1)
            {
                currentpage = 3;//零食
                mouse_off_arrow(&mouse);//隐藏鼠标
```

```
                draw_user_shop(products, productCount,currentpage);//
绘制用户超市页面
                mouse_on_arrow(mouse);//显示鼠标
            }
            else if(mouse_press(560, 0, 680, 50)==1)
            {
                currentpage = 4;//饮料
                mouse_off_arrow(&mouse);//隐藏鼠标
                draw_user_shop(products, productCount,currentpage);//
绘制用户超市页面
                mouse_on_arrow(mouse);//显示鼠标
            }
            else if(mouse_press(680, 0, 800, 50)==1)
            {
                currentpage = 5;//运动用品
                mouse_off_arrow(&mouse);//隐藏鼠标
                draw_user_shop(products, productCount,currentpage);//
绘制用户超市页面
                mouse_on_arrow(mouse);//显示鼠标
            }
            else if(mouse_press(800, 0, 920, 50)==1)
            {
                currentpage = 6;//水果
                mouse_off_arrow(&mouse);//隐藏鼠标
                draw_user_shop(products, productCount,currentpage);//
绘制用户超市页面
                mouse_on_arrow(mouse);//显示鼠标
            }
            else if(mouse_press(920, 0, 1024, 50)==1)
            {
                currentpage = 7;//文创
                mouse_off_arrow(&mouse);//隐藏鼠标
                draw_user_shop(products, productCount,currentpage);//
```

```
                mouse_on_arrow(mouse);//显示鼠标
            }
            else if(mouse_press(270, 235, 1070, 835)==1)//点击商品
            {
                MouseGet(&mouse);
                AddSub(mouse.x, mouse.y, productCount, products, carts,
&cart.itemCount, currentpage - 1); //注意 currentpage 从 1 开始,计算 index
时需减 1
                     draw_user_shop_quantity(products,  productCount,
currentpage); //刷新页面显示更新后的数量
                delay(100);
            }

            //点击排序
            if (state == 1) {
                if (mouse_press(205, 95, 445, 144) == 1) // 点击从高到
低
                {
                    int i, j;
                    int cnt = (currentpage - 1) * 12; // 当前页起始下标
                    int end = cnt + 12;

                    for (i = cnt; i < end - 1; i++) {
                        for (j = cnt; j < end - 1 - (i - cnt); j++) {
                                if (products[j].price < products[j +
1].price) {
                                    Product temp = products[j];
                                    products[j] = products[j + 1];
                                    products[j + 1] = temp;
                                }
                        }
                    }
```

```
                mouse_off_arrow(&mouse);//隐藏鼠标
                                                draw_user_shop(products,
productCount,currentpage);//绘制用户超市页面
                mouse_on_arrow(mouse);//显示鼠标
                state=0;
            }
             else if(mouse_press(205, 146, 445, 295)==1)//点击从低
到高
            {
                int i, j;
                int cnt = (currentpage - 1) * 12; // 当前页起始下标
                int end = cnt + 12;

                for (i = cnt; i < end - 1; i++) {
                    for (j = cnt; j < end - 1 - (i - cnt); j++) {
                             if (products[j].price > products[j +
1].price) {
                                Product temp = products[j];
                                products[j] = products[j + 1];
                                products[j + 1] = temp;
                            }
                    }
                }

                mouse_off_arrow(&mouse);//隐藏鼠标
                                                draw_user_shop(products,
productCount,currentpage);//绘制用户超市页面
                mouse_on_arrow(mouse);//显示鼠标
                state=0;
            }

        }
```

```
        }
    }

    //绘制用户超市页面
    void   draw_user_shop(Product   products[],int   productCount,int
currentpage){
        int i=0;
        int j=0;
        int cnt=0;

        bar1(200, 0, 1024, 768,white);

        Line_Thick(200,50,1024,50,2,deepblue);

        Line_Thick(320,0,320,50,2,deepblue);
        Line_Thick(440,0,440,50,2,deepblue);
        Line_Thick(560,0,560,50,2,deepblue);
        Line_Thick(680,0,680,50,2,deepblue);
        Line_Thick(800,0,800,50,2,deepblue);
        Line_Thick(920,0,920,50,2,deepblue);

        Draw_Rounded_Rectangle(800, 700, 1000, 750, 5,1,deepblue);//
购物车按钮

        Line_Thick(220,75,235,90,1,black);//
        Line_Thick(235,90,250,75,1,black);//


        PrintCC(860,715,"购物车",HEI,24,1,deepblue);
        PrintCC(220,700,"喻园超市",HEI,48,1,deepblue);

        //显示不同商品类型
        switch (currentpage)
```

```
{
    case 1:{
        PrintCC(210,15,"生活用品",HEI,24,1,deepblue);
        PrintCC(355,15,"文具",HEI,24,1,grey);
        PrintCC(475,15,"零食",HEI,24,1,grey);
        PrintCC(595,15,"饮料",HEI,24,1,grey);
        PrintCC(690,15,"运动用品",HEI,24,1,grey);
        PrintCC(835,15,"水果",HEI,24,1,grey);
        PrintCC(955,15,"文创",HEI,24,1,grey);
        break;
    }
    case 2:{
        PrintCC(210,15,"生活用品",HEI,24,1,grey);
        PrintCC(355,15,"文具",HEI,24,1,deepblue);
        PrintCC(475,15,"零食",HEI,24,1,grey);
        PrintCC(595,15,"饮料",HEI,24,1,grey);
        PrintCC(690,15,"运动用品",HEI,24,1,grey);
        PrintCC(835,15,"水果",HEI,24,1,grey);
        PrintCC(955,15,"文创",HEI,24,1,grey);
        break;
    }
    case 3:{
        PrintCC(210,15,"生活用品",HEI,24,1,grey);
        PrintCC(355,15,"文具",HEI,24,1,grey);
        PrintCC(475,15,"零食",HEI,24,1,deepblue);
        PrintCC(595,15,"饮料",HEI,24,1,grey);
        PrintCC(690,15,"运动用品",HEI,24,1,grey);
        PrintCC(835,15,"水果",HEI,24,1,grey);
        PrintCC(955,15,"文创",HEI,24,1,grey);
        break;
    }
    case 4:{
        PrintCC(210,15,"生活用品",HEI,24,1,grey);
```

```
    PrintCC(355,15,"文具",HEI,24,1,grey);
    PrintCC(475,15,"零食",HEI,24,1,grey);
    PrintCC(595,15,"饮料",HEI,24,1,deepblue);
    PrintCC(690,15,"运动用品",HEI,24,1,grey);
    PrintCC(835,15,"水果",HEI,24,1,grey);
    PrintCC(955,15,"文创",HEI,24,1,grey);
    break;
}
case 5:{
    PrintCC(210,15,"生活用品",HEI,24,1,grey);
    PrintCC(355,15,"文具",HEI,24,1,grey);
    PrintCC(475,15,"零食",HEI,24,1,grey);
    PrintCC(595,15,"饮料",HEI,24,1,grey);
    PrintCC(690,15,"运动用品",HEI,24,1,deepblue);
    PrintCC(835,15,"水果",HEI,24,1,grey);
    PrintCC(955,15,"文创",HEI,24,1,grey);
    break;
}
case 6:{
    PrintCC(210,15,"生活用品",HEI,24,1,grey);
    PrintCC(355,15,"文具",HEI,24,1,grey);
    PrintCC(475,15,"零食",HEI,24,1,grey);
    PrintCC(595,15,"饮料",HEI,24,1,grey);
    PrintCC(690,15,"运动用品",HEI,24,1,grey);
    PrintCC(835,15,"水果",HEI,24,1,deepblue);
    PrintCC(955,15,"文创",HEI,24,1,grey);
    break;
}
case 7:{
    PrintCC(210,15,"生活用品",HEI,24,1,grey);
    PrintCC(355,15,"文具",HEI,24,1,grey);
    PrintCC(475,15,"零食",HEI,24,1,grey);
    PrintCC(595,15,"饮料",HEI,24,1,grey);
```

```
            PrintCC(690,15,"运动用品",HEI,24,1,grey);
            PrintCC(835,15,"水果",HEI,24,1,grey);
            PrintCC(955,15,"文创",HEI,24,1,deepblue);
            break;
        }
    }

    cnt=(currentpage-1)*12;//决定显示哪一页的商品

    //显示商品信息
    for(j=0;j<3;j++){
        for(i=0;i<4;i++){//先横向，再竖向
            char quantity_str[20];
            char price_str[20];
            // 使用 sprintf 将 quantity 转换为字符串
            sprintf(quantity_str, "%d", products[cnt].quantity);
            sprintf(price_str, "%.2f", products[cnt].price);

            Line_Thick(270+200*i, 235+200*j, 290+200*i, 235+200*j,
1, deepblue);//减号

            Line_Thick(390+200*i, 235+200*j, 370+200*i, 235+200*j,
1, deepblue);//加号
            Line_Thick(380+200*i, 225+200*j, 380+200*i, 245+200*j,
1, deepblue);


             PrintCC(270+200*i,75+200*j,products[cnt].name,HEI,24,
1,black);//显示商品名称

              PrintText(270+22+200*i, 220+3+200*j,price_str, HEI,
24, 0, black);//显示商品价格
                        PrintText(395+200*i,75+200*j,   (unsigned
```

```
char*)quantity_str,HEI,24,0,black);//显示商品数量


                                    Readbmp64k(270+200*i,    100+200*j,
products[cnt].photo);//显示商品图片
                cnt++;

            }
        }

    }

    //重新显示商品数量
    void        draw_user_shop_quantity(Product        products[],int
productCount,int currentpage){
        int i=0;
        int j=0;
        int cnt=0;
        cnt=(currentpage-1)*12;//决定显示哪一页的商品

        //显示商品信息
        for(j=0;j<3;j++){
            for(i=0;i<4;i++){//先横向，再竖向
                char quantity_str[20];
                // 使用 sprintf 将 quantity 转换为字符串
                sprintf(quantity_str, "%d", products[cnt].quantity);
                 // 调用 PrintText 函数，将 quantity_str 转换为 unsigned
char* 类型
                bar1(395+200*i, 75+200*j, 460+200*i, 95+200*j,white);
                            PrintText(395+200*i,75+200*j,   (unsigned
char*)quantity_str,HEI,24,0,black);//显示商品数量
                cnt++;
```

```
            }
        }
    }

    //加减商品
    void AddSub(int mx, int my, int productCount, Product products[],
CartItem carts[], int* itemCount, int currentPage) {
        int i, j, index;
        int baseX = 270, baseY = 235;
        int width = 200, height = 200;

        for (j = 0; j < 3; j++) {
            for (i = 0; i < 4; i++) {
                index = currentPage * 12 + i + j * 4;
                if (index >= productCount) return;

                // 加号区域
                if (mx >= 370 + i * width && mx <= 390 + i * width &&
                    my >= 225 + j * height && my <= 245 + j * height)
{
                    addToCart(products[index], carts, itemCount,index);
                    products[index].quantity++;
                    return;
                }
                // 减号区域
                if (mx >= 270 + i * width && mx <= 290 + i * width &&
                    my >= 225 + j * height && my <= 245 + j * height)
{
                    if (products[index].quantity > 0) {
                        products[index].quantity--;  // 商品页面数量 -1
                               removeFromCart(products[index],  carts,
itemCount);
                    }else {
```

```
                                PrintCC(220,660,"此商品已从购物车中移除
",HEI,24,1,lightred);
                        delay(500);
                        bar1(220,660,800,700,white);
                    }
                    return;
                }
            }
        }
    }

    void addToCart(Product p, CartItem carts[], int *itemCount,int
index) {
        int i=0;
        for (i = 0; i < *itemCount; i++) {
            if (carts[i].id == p.id) {
                carts[i].quantity++;
                return;
            }
        }
        carts[*itemCount].id = p.id;
        strcpy(carts[*itemCount].name, p.name);
        strcpy(carts[*itemCount].photo, p.photo);
        carts[*itemCount].price=p.price;
        carts[*itemCount].quantity = 1;
        carts[*itemCount].index_in_products=index;//在商品数组中的索引
        (*itemCount)++;
    }

    void removeFromCart(Product p, CartItem carts[], int *itemCount)
{
        int i, j;
        for (i = 0; i < *itemCount; i++) {
```

```
            if (carts[i].id == p.id) {
                if (carts[i].quantity > 1) {
                    carts[i].quantity--;
                } else if (carts[i].quantity == 1) {
                    // 如果减到 0，删除这个商品
                    for (j = i; j < *itemCount - 1; j++) {
                        carts[j] = carts[j + 1];
                    }
                    (*itemCount)--;
                }
                return;
            }
        }
    }

    void draw_sort(){
        Fill_Rounded_Rectangle(205, 95, 455, 200, 30,snow);//填色
        Draw_Rounded_Rectangle(205, 95, 455, 200, 30, 1,0x6B4D);//最外
围灰色圆角矩形

        PrintText(225, 110,"价格从高到低排序",HEI,24,1,black);
        Line_Thick(215, 145, 445, 145, 1, deepgrew);//横线
        PrintText(225, 160,"价格从低到高排序",HEI,24,1,black);
    }
/*****************************************************************/
    #include "all_func.h"

    Canteen canteen[17]={
        {1,"韵苑学生食堂"},
        {2,"东园食堂"},
        {3,"学一食堂"},
        {4,"学二食堂"},
        {5,"东教工食堂"},
```

```
    {6,"喻园食堂"},
    {7,"集贤楼食堂"},
    {8,"东一食堂"},
    {9,"紫荆园餐厅"},
    {10,"东三食堂"},
    {11,"东四食堂"},
    {12,"西一食堂"},
    {13,"西二食堂"},
    {14,"西三食堂"},
    {15,"百景园餐厅"},
    {16,"集锦园餐厅"},
    {17,"百惠园餐厅"}
};
void user_takeout(){
    int index=0;
    int mx=0;
    int my=0;

    mouse_off_arrow(&mouse);

    draw_user_takeout();

    mouse_on_arrow(mouse);

    while(1){
        mouse_show_arrow(&mouse);

        if(mouse_press(40, 113, 160, 163)==1)
        {
            return;
            //welcome();//首页
        }
        else if(mouse_press(40, 276, 160, 326)==1)
```

```
    {
        press_func(1);//进入超市页面
        user_shop();//用户超市页面
        return;
    }
    else if(mouse_press(40, 439, 160, 489)==1)
    {
        press_func(2);//进入外卖页面
        user_takeout();//用户外卖页面
        return;
    }
    else if(mouse_press(40, 602, 160, 652)==1)
    {
        press_func(3);//进入快递页面
        user_deliver();//用户快递页面
        return;
    }else if(mouse_press(200, 0, 1024, 768)==1)//选择食堂
    {
        MouseGet(&mouse);
        mx=mouse.x;
        my=mouse.y;
        index=press_canteen(mx,my);//获取食堂编号
        user_food(index);//进入菜品页面

        //return 后从这开始
        mouse_off_arrow(&mouse);
        bar1(200, 0, 1024, 768, white); // 清除注册界面残留
        draw_user_takeout();
        mouse_on_arrow(mouse);

    }

}
```

```
    }
    void draw_user_takeout(){
        int i,j;
        int cnt=0;
        bar1(200, 0, 1024, 768,white);

        PrintCC(250,50,"请选择食堂",HEI,24,1,deepblue);

        for(i=0;i<6;i++){
            for(j=0;j<3;j++){
            Draw_Rounded_Rectangle(250+250*j, 120+80*i, 250+250*j+185,
120+80*i+50, 5,1,0x0235);
            PrintCC(250+250*j+17,120+80*i+13,canteen[cnt].name,HEI,24
,1,0x0235);
            cnt++;
            }
        }

           //   Draw_Rounded_Rectangle(250,   120,   250+185,   120+50,
5,1,0x0235);
        // PrintCC(250+17,120+13,"韵苑学生食堂",HEI,24,1,0x0235);

           //   Draw_Rounded_Rectangle(500,   120,   500+185,   120+50,
5,1,0x0235);
        // PrintCC(500+17,120+13,"东园食堂",HEI,24,1,0x0235);

           //   Draw_Rounded_Rectangle(750,   120,   750+185,   120+50,
5,1,0x0235);
        // PrintCC(750+17,120+13,"东教工食堂",HEI,24,1,0x0235);

           //   Draw_Rounded_Rectangle(250,   200,   250+185,   200+50,
5,1,0x0235);
        // PrintCC(250+17,200+13,"学生一食堂",HEI,24,1,0x0235);
```

```
          //   Draw_Rounded_Rectangle(500,   200,   500+185,   200+50,
5,1,0x0235);
       // PrintCC(500+17,200+13,"学生二食堂",HEI,24,1,0x0235);

          //   Draw_Rounded_Rectangle(750,   200,   750+185,   200+50,
5,1,0x0235);
       // PrintCC(750+17,200+13,"紫荆园餐厅",HEI,24,1,0x0235);

          //   Draw_Rounded_Rectangle(250,   280,   250+185,   280+50,
5,1,0x0235);
       // PrintCC(250+17,280+13,"东一食堂",HEI,24,1,0x0235);

          //   Draw_Rounded_Rectangle(500,   280,   500+185,   280+50,
5,1,0x0235);
       // PrintCC(500+17,280+13,"东三食堂",HEI,24,1,0x0235);

          //   Draw_Rounded_Rectangle(750,   280,   750+185,   280+50,
5,1,0x0235);
       // PrintCC(750+17,280+13,"喻园餐厅",HEI,24,1,0x0235);

          //   Draw_Rounded_Rectangle(250,   360,   250+185,   360+50,
5,1,0x0235);
       // PrintCC(250+17,360+13,"百景园",HEI,24,1,0x0235);

          //   Draw_Rounded_Rectangle(500,   360,   500+185,   360+50,
5,1,0x0235);
       // PrintCC(500+17,360+13,"西一食堂",HEI,24,1,0x0235);

          //   Draw_Rounded_Rectangle(750,   360,   750+185,   360+50,
5,1,0x0235);
       // PrintCC(750+17,360+13,"西二食堂",HEI,24,1,0x0235);
```

```
            //   Draw_Rounded_Rectangle(250,   440,   250+185,   440+50,
5,1,0x0235);
        // PrintCC(250+17,440+13,"东园食堂",HEI,24,1,0x0235);

            //   Draw_Rounded_Rectangle(500,   440,   500+185,   440+50,
5,1,0x0235);
        // PrintCC(500+17,440+13,"东教工食堂",HEI,24,1,0x0235);

            //   Draw_Rounded_Rectangle(750,   440,   750+185,   440+50,
5,1,0x0235);
        // PrintCC(750+17,440+13,"西园食堂",HEI,24,1,0x0235);

            //   Draw_Rounded_Rectangle(250,   520,   250+185,   520+50,
5,1,0x0235);
        // PrintCC(250+17,520+13,"南园食堂",HEI,24,1,0x0235);

            //   Draw_Rounded_Rectangle(500,   520,   500+185,   520+50,
5,1,0x0235);
        // PrintCC(500+17,520+13,"中心食堂",HEI,24,1,0x0235);

            //   Draw_Rounded_Rectangle(750,   520,   750+185,   520+50,
5,1,0x0235);
        // PrintCC(750+17,520+13,"韵苑食堂",HEI,24,1,0x0235);
    }

    int press_canteen(int mx, int my){
        //if(mx < 250 || mx > 935 || my < 120 || my > 570) return; //
边界检查

        int row = (my - 120) / 80;
        int col = (mx - 250) / 250;

        // 每个按钮的精确区域
```

```
        int btn_x = 250 + col * 250;
        int btn_y = 120 + row * 80;

        if(mx >= btn_x && mx <= btn_x + 185 && my >= btn_y && my <=
btn_y + 50){
            int index = row * 3 + col + 1;// 计算索引
        }
    }
/*****************************************************************/
    #include "all_func.h"

    void user(int user_pos){

        int cur_index = -1;//
        int cur_community=0;
        int returned_index;

        UserList UL = {0};
        USER currentUser;

        ReadAllUser(&UL); // 读取用户列表

        currentUser=UL.elem[user_pos];// 获取当前用户信息

        DestroyUList(&UL); // 释放用户列表空间

        mouse_off_arrow(&mouse);

        draw_user();

        mouse_on_arrow(mouse);

        while(1){
```

```
mouse_show_arrow(&mouse);

if(mouse_press(40, 113, 160, 163)==1)
{
    return;
    //welcome();//首页
}
else if(mouse_press(40, 276, 160, 326)==1)
{
    press_func(1);//进入超市页面
    user_shop();//用户超市页面

    //return 后从这开始
    mouse_off_arrow(&mouse);
    bar1(200, 0, 1024, 768, white); // 清除超市界面残留
    draw_user();
    mouse_on_arrow(mouse);
}
else if(mouse_press(40, 439, 160, 489)==1)
{
    press_func(2);//进入外卖页面
    user_takeout();//用户外卖页面

    //return 后从这开始
    mouse_off_arrow(&mouse);
    bar1(200, 0, 1024, 768, white); // 清除超市界面残留
    draw_user();
    mouse_on_arrow(mouse);
}
else if(mouse_press(40, 602, 160, 652)==1)
{
    press_func(3);//进入快递页面
    user_deliver();//用户快递页面
```

```
            //return 后从这开始
            mouse_off_arrow(&mouse);
            bar1(200, 0, 1024, 768, white); // 清除超市界面残留
            draw_user();
            mouse_on_arrow(mouse);
        }
        else if(mouse_press(430, 105, 650, 155)==1)
        {
            number_input(currentUser.number, 435, 110, 645, 150);
// 输入手机号
        }
        else if(mouse_press(710, 105, 830, 155)==1)
        {
             if(strlen(currentUser.number)==11)//判断手机号长度是否
合法
            {
                save_user(currentUser);
                PrintCC(250,260,"保存成功",HEI,24,1,lightred);
                delay(500);
                bar1(250,260,400,310,white);
            }
            else
            {
                PrintCC(250,260,"长度不合法",HEI,24,1,lightred);
                delay(500);
                bar1(250,260,400,310,white);
            }
        }
        else if(mouse_press(440, 180, 560, 230)==1)
        {
            cur_index = -1;
            press_func(4);//东区
```

```
    draw_button(1);
    cur_community=1;

}
else if(mouse_press(620, 180, 740, 230)==1)
{
    cur_index = -1;
    press_func(5);//西区
    draw_button(2);
    cur_community=2;
}
else if(mouse_press(800, 180, 920, 230)==1)
{
    cur_index = -1;
    press_func(6);//南区
    draw_button(3);
    cur_community=3;

}
else if(mouse_press(530, 255, 650, 305)==1)
{
    cur_index = -1;
    press_func(7);//紫菘
    draw_button(4);
    cur_community=4;
}
else if(mouse_press(750, 255, 870, 305)==1)
{
    cur_index = -1;
    press_func(8);//韵苑
    draw_button(5);
    cur_community=5;
}
```

```
            else if (mouse_press(200, 310, 1024, 768) == 1) {

                MouseGet(&mouse);
                mouse_off_arrow(&mouse);
                     returned_index = press_button(mouse.x, mouse.y,
cur_index, cur_community);//获取按钮编号

                 if(returned_index>=0)//如果返回值大于等于 0,则说明选择了
按钮
                {
                                                currentUser.community    =
button[returned_index].commmunity;//获取社区编号

                                                    currentUser.index    =
button[returned_index].index;//获取楼号编号

                    save_user(currentUser);//保存用户信息
                }

                cur_index = returned_index;//更新当前按钮编号

                mouse_on_arrow(mouse);

                delay(200);
            }
        }
    }

    //绘制用户界面
    void draw_user()
    {
        bar1(0, 0, 1024, 768,white);
        bar1(0, 0, 200, 768,deepblue);//左侧背景
```

```
        Readbmp64k(597, 0, "bmp\\hust1.bmp");//背景图

        Fill_Rounded_Rectangle(40, 113, 160, 163, 25,white);//填色
        Fill_Rounded_Rectangle(40, 276, 160, 326, 25,white);//填色
        Fill_Rounded_Rectangle(40, 439, 160, 489, 25,white);//填色
        Fill_Rounded_Rectangle(40, 602, 160, 652, 25,white);//填色

        Draw_Rounded_Rectangle(40, 113, 160, 163, 25, 1,deepblue);//
返回按钮
        Draw_Rounded_Rectangle(40, 276, 160, 326, 25, 1,deepblue);//
超市按钮
        Draw_Rounded_Rectangle(40, 439, 160, 489, 25, 1,deepblue);//
外卖按钮
        Draw_Rounded_Rectangle(40, 602, 160, 652, 25, 1,deepblue);//
快递按钮
        Draw_Rounded_Rectangle(440, 180, 560, 230, 25, 1,deepblue);//
东区按钮
        Draw_Rounded_Rectangle(620, 180, 740, 230, 25, 1,deepblue);//
西区按钮
        Draw_Rounded_Rectangle(800, 180, 920, 230, 25, 1,deepblue);//
南区按钮
        Draw_Rounded_Rectangle(530, 255, 650, 305, 25, 1,deepblue);//
紫菘按钮
        Draw_Rounded_Rectangle(750, 255, 870, 305, 25, 1,deepblue);//
韵苑按钮

        Draw_Rounded_Rectangle(430, 105, 650, 155, 5, 1,deepblue);//
手机号输入框
        Draw_Rounded_Rectangle(710, 105, 830, 155, 25, 1,deepblue);//
保存按钮

        PrintCC(75,128,"返回",HEI,24,1,deepblue);
        PrintCC(75,291,"超市",HEI,24,1,deepblue);
```

```
        PrintCC(75,454,"外卖",HEI,24,1,deepblue);
        PrintCC(75,617,"快递",HEI,24,1,deepblue);
        PrintCC(475,195,"东区",HEI,24,1,deepblue);
        PrintCC(655,195,"西区",HEI,24,1,deepblue);
        PrintCC(835,195,"南区",HEI,24,1,deepblue);
        PrintCC(565,270,"紫菘",HEI,24,1,deepblue);
        PrintCC(785,270,"韵苑",HEI,24,1,deepblue);
        PrintCC(745,120,"保存",HEI,24,1,deepblue);

        PrintCC(250,50,"当前账号类型为：用户",HEI,24,1,deepblue);
        PrintCC(250,120,"请输入手机号：",HEI,24,1,deepblue);
        PrintCC(250,190,"请选择住址：",HEI,24,1,deepblue);
    }

    void press_func(int x){
        mouse_off_arrow(&mouse);
        switch (x)
        {
            case 1:{
                Fill_Rounded_Rectangle(40, 276, 160, 326, 25,deepblue);
                     Draw_Rounded_Rectangle(40,  276,  160,  326,  25,
1,deepblue);
                PrintCC(75,291,"超市",HEI,24,1,white);
                Fill_Rounded_Rectangle(40, 439, 160, 489, 25,white);
                     Draw_Rounded_Rectangle(40,  439,  160,  489,  25,
1,deepblue);
                PrintCC(75,454,"外卖",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(40, 602, 160, 652, 25,white);
                     Draw_Rounded_Rectangle(40,  602,  160,  652,  25,
1,deepblue);
                PrintCC(75,617,"快递",HEI,24,1,deepblue);
                break;
            }
```

```
            case 2:{
                Fill_Rounded_Rectangle(40, 276, 160, 326, 25,white);
                     Draw_Rounded_Rectangle(40,  276,  160,  326,  25,
1,deepblue);
                PrintCC(75,291,"超市",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(40, 439, 160, 489, 25,deepblue);
                     Draw_Rounded_Rectangle(40,  439,  160,  489,  25,
1,deepblue);
                PrintCC(75,454,"外卖",HEI,24,1,white);
                Fill_Rounded_Rectangle(40, 602, 160, 652, 25,white);
                     Draw_Rounded_Rectangle(40,  602,  160,  652,  25,
1,deepblue);
                PrintCC(75,617,"快递",HEI,24,1,deepblue);
                break;
            }
            case 3:{
                Fill_Rounded_Rectangle(40, 276, 160, 326, 25,white);
                     Draw_Rounded_Rectangle(40,  276,  160,  326,  25,
1,deepblue);
                PrintCC(75,291,"超市",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(40, 439, 160, 489, 25,white);
                     Draw_Rounded_Rectangle(40,  439,  160,  489,  25,
1,deepblue);
                PrintCC(75,454,"外卖",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(40, 602, 160, 652, 25,deepblue);
                     Draw_Rounded_Rectangle(40,  602,  160,  652,  25,
1,deepblue);
                PrintCC(75,617,"快递",HEI,24,1,white);
                break;
            }
            case 4:{
                     Fill_Rounded_Rectangle(440,  180,  560,  230,  25,
deepblue);
```

```
                Draw_Rounded_Rectangle(440, 180, 560, 230, 25,
1,white);
            PrintCC(475,195,"东区",HEI,24,1,white);
            Fill_Rounded_Rectangle(620, 180, 740, 230, 25, white);
                Draw_Rounded_Rectangle(620, 180, 740, 230, 25,
1,deepblue);
            PrintCC(655,195,"西区",HEI,24,1,deepblue);
            Fill_Rounded_Rectangle(800, 180, 920, 230, 25, white);
                Draw_Rounded_Rectangle(800, 180, 920, 230, 25,
1,deepblue);
            PrintCC(835,195,"南区",HEI,24,1,deepblue);
            Fill_Rounded_Rectangle(530, 255, 650, 305, 25, white);
                Draw_Rounded_Rectangle(530, 255, 650, 305, 25,
1,deepblue);
            PrintCC(565,270,"紫菘",HEI,24,1,deepblue);
            Fill_Rounded_Rectangle(750, 255, 870, 305, 25, white);
                Draw_Rounded_Rectangle(750, 255, 870, 305, 25,
1,deepblue);
            PrintCC(785,270,"韵苑",HEI,24,1,deepblue);
            break;
        }
        case 5:{
            Fill_Rounded_Rectangle(440, 180, 560, 230, 25, white);
                Draw_Rounded_Rectangle(440, 180, 560, 230, 25,
1,deepblue);
            PrintCC(475,195,"东区",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(620, 180, 740, 230, 25,
deepblue);
                Draw_Rounded_Rectangle(620, 180, 740, 230, 25,
1,white);
            PrintCC(655,195,"西区",HEI,24,1,white);
            Fill_Rounded_Rectangle(800, 180, 920, 230, 25, white);
                Draw_Rounded_Rectangle(800, 180, 920, 230, 25,
```

```
1,deepblue);
                PrintCC(835,195,"南区",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(530, 255, 650, 305, 25, white);
                     Draw_Rounded_Rectangle(530,  255,  650,  305,  25,
1,deepblue);
                PrintCC(565,270,"紫菘",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(750, 255, 870, 305, 25, white);
                     Draw_Rounded_Rectangle(750,  255,  870,  305,  25,
1,deepblue);
                PrintCC(785,270,"韵苑",HEI,24,1,deepblue);
                break;
            }
            case 6:{
                Fill_Rounded_Rectangle(440, 180, 560, 230, 25, white);
                     Draw_Rounded_Rectangle(440,  180,  560,  230,  25,
1,deepblue);
                PrintCC(475,195,"东区",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(620, 180, 740, 230, 25, white);
                     Draw_Rounded_Rectangle(620,  180,  740,  230,  25,
1,deepblue);
                PrintCC(655,195,"西区",HEI,24,1,deepblue);
                     Fill_Rounded_Rectangle(800,  180,  920,  230,  25,
deepblue);
                     Draw_Rounded_Rectangle(800,  180,  920,  230,  25,
1,white);
                PrintCC(835,195,"南区",HEI,24,1,white);
                Fill_Rounded_Rectangle(530, 255, 650, 305, 25, white);
                     Draw_Rounded_Rectangle(530,  255,  650,  305,  25,
1,deepblue);
                PrintCC(565,270,"紫菘",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(750, 255, 870, 305, 25, white);
                     Draw_Rounded_Rectangle(750,  255,  870,  305,  25,
1,deepblue);
```

```
                PrintCC(785,270,"韵苑",HEI,24,1,deepblue);
                break;
            }
            case 7 :{
                Fill_Rounded_Rectangle(440, 180, 560, 230, 25, white);
                    Draw_Rounded_Rectangle(440,  180,  560,  230,  25,
1,deepblue);
                PrintCC(475,195,"东区",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(620, 180, 740, 230, 25, white);
                    Draw_Rounded_Rectangle(620,  180,  740,  230,  25,
1,deepblue);
                PrintCC(655,195,"西区",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(800, 180, 920, 230, 25, white);
                    Draw_Rounded_Rectangle(800,  180,  920,  230,  25,
1,deepblue);
                PrintCC(835,195,"南区",HEI,24,1,deepblue);
                    Fill_Rounded_Rectangle(530,  255,  650,  305,  25,
deepblue);
                    Draw_Rounded_Rectangle(530,  255,  650,  305,  25,
1,white);
                PrintCC(565,270,"紫菘",HEI,24,1,white);
                Fill_Rounded_Rectangle(750, 255, 870, 305, 25, white);
                    Draw_Rounded_Rectangle(750,  255,  870,  305,  25,
1,deepblue);
                PrintCC(785,270,"韵苑",HEI,24,1,deepblue);
                break;
            }
            case 8:{
                Fill_Rounded_Rectangle(440, 180, 560, 230, 25, white);
                    Draw_Rounded_Rectangle(440,  180,  560,  230,  25,
1,deepblue);
                PrintCC(475,195,"东区",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(620, 180, 740, 230, 25, white);
```

```
                    Draw_Rounded_Rectangle(620,  180,  740,  230,  25,
1,deepblue);
                PrintCC(655,195,"西区",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(800, 180, 920, 230 ,25 ,white);
                Draw_Rounded_Rectangle(800 ,180 ,920 ,230 ,25 ,1 ,dee
pblue);
                PrintCC(835 ,195 ,"南区" ,HEI ,24 ,1 ,deepblue);
                Fill_Rounded_Rectangle(530 ,255 ,650 ,305 ,25 ,white);
                Draw_Rounded_Rectangle(530 ,255 ,650 ,305 ,25 ,1 ,dee
pblue);
                PrintCC(565 ,270 ,"紫菘" ,HEI ,24 ,1 ,deepblue);
                Fill_Rounded_Rectangle(750 ,255 ,870 ,305 ,25 ,deepbl
ue);
                Draw_Rounded_Rectangle(750 ,255 ,870 ,305 ,25 ,1 ,whi
te);
                PrintCC(785 ,270 ,"韵苑" ,HEI ,24 ,1 ,white);
                break;
            }
            case 9:{
                    Draw_Rounded_Rectangle(440,  180,  560,  230,  25,
1,deepblue);
                PrintCC(475,195,"东区",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(620, 180, 740, 230, 25, white);
                    Draw_Rounded_Rectangle(620,  180,  740,  230,  25,
1,deepblue);
                PrintCC(655,195,"西区",HEI,24,1,deepblue);
                Fill_Rounded_Rectangle(800, 180, 920, 230 ,25 ,white);
                Draw_Rounded_Rectangle(800 ,180 ,920 ,230 ,25 ,1 ,dee
pblue);
                PrintCC(835 ,195 ,"南区" ,HEI ,24 ,1 ,deepblue);
                Fill_Rounded_Rectangle(530 ,255 ,650 ,305 ,25 ,white);
                Draw_Rounded_Rectangle(530 ,255 ,650 ,305 ,25 ,1 ,dee
pblue);
```

```
                PrintCC(565 ,270 ,"紫菘" ,HEI ,24 ,1 ,deepblue);
                Fill_Rounded_Rectangle(750 ,255 ,870 ,305 ,25 ,deepbl
ue);
                Draw_Rounded_Rectangle(750 ,255 ,870 ,305 ,25 ,1 ,whi
te);
                PrintCC(785 ,270 ,"韵苑" ,HEI ,24 ,1 ,white);
                break;
            }
            default:
                break;
        }
        mouse_on_arrow(mouse);
    }

    //输入手机号
    void   number_input(char   *number,int   bar_x1,int   bar_y1,int
bar_x2,int bar_y2)
    {
        int length;
        char number_temp[12]={'\0'};//重新进入该页面时输入框清零
        char showtemp[2]= "\0";//存储输入字符,用于输入框展示
         int  i=0,k,temp;   //  i 为字符个数,temp 为从键盘上读取输入字符的
ACSII 码
        int border; //光标的横坐标
        int x1,y1;
        x1=bar_x1+4;
        y1=bar_y1+5;//光标相较于输入框的偏移量
        border=x1+4;//每个字符占 8 个像素,每输入一个字符光标右移 8 个像素

        if(number_temp[0]=='\0') //如果账号为空，则显示输入框
            bar1(bar_x1, bar_y1, bar_x2, bar_y2,0xFFFF);
        else
        {            //光标定位至文本末尾
```

```
            length=strlen(number_temp);
            i=length;
            border+=12*i;
            cursor(border,y1);
        }

        while(1)
        {
            cursor(border,y1);
                        if(mouse_location(455,255,845,295)==1    &&
mouse_location(455,335,845,375)==1                                &&
mouse_location(455,415,845,455)==1)
                mouse_show_cursor(&mouse);
            else
                mouse_show_arrow(&mouse);
            if(bioskey(1)) //如果有键盘输入
            {
                temp=bioskey(0)&0x00ff; //获取键盘输入
                  if(temp!='\r'&&temp!='\n')  //检测输入不为回车键，则继
续，否则输入结束
                {
                     if(('0'<=temp&&temp<='9')&& i <11)//检测为数字或字
母，则记录
                    {
                        hide_cursor(border,y1); //隐藏原光标

                         number_temp[i]=temp;//字符送入给定字符串，用于保
存用户信息
                        number[i]=temp;

                        *showtemp=temp;  //temp 转化为字符串
                        PrintText(border,y1+2,showtemp,HEI,24,1,0); //
显示新的字符串达到画面与实际输入的同步
```

```
            i++;    //字符个数自增

            number_temp[i]='\0';//标记字符串结尾
            number[i]='\0';

            border+=12; //光标横坐标右移 12 像素
            draw_cursor(border,y1);
        }
        else if(temp=='\b'&&i>0)  //检测是否为退格键，是则消除前一个字符
        {
            hide_cursor(border,y1); //隐藏原光标
            border-=12; //光标左移 12 像素
            i--;    //字符个数自减

            number_temp[i]='\0';//将存储的字符用 0 覆盖
            number[i]='\0';

            bar1(border,y1,border+10, y1+24, 0xffff);   //清空原字符

            draw_cursor(border,y1);
        }
        else if(i>=11)
        {
            mouse_off_arrow(&mouse);
            hide_cursor(border,y1); //隐藏原光标
            break;//超出 11 字符就退出
        }
    }
    else
    {
        break;
    }
```

```
            }
            else if (mouse_press(bar_x1,bar_y1,bar_x2,bar_y2)==2)  //
点击框外
            {
                hide_cursor(border,y1);//隐藏光标
                break;
            }

        // hide_cursor(border,y1);  //隐藏光标
        }
    }
/*****************************************************************/
    #include "all_func.h"

    int main()
    {

        int gd = DETECT, gm;
        int distance;
        distance = dijkstra(&node[10], &node[344],2);//测试 dijkstra 算
法
        SetSVGA64k();//启动 SVGA 画图界面

        mouse_init(); // 初始化鼠标

        mouse_on_arrow(mouse);

        while (1)
        {
            mouse_show_arrow(&mouse);// 更新鼠标位置

            welcome();//首页
```

```
        }
        CloseSVGA();//关闭图形界面
        printf("Distance: %d\n", distance); // 打印距离
        return 0;
    }
/*****************************************************************/
    #include "all_func.h"

    int welcome() {

        int i;//循环变量

        char name[12]="\0";//用户名
        char code[12]="\0";//密码
        char judge[12]="\0";//用于判断的密码
        int result=-5;//判断登陆结果,-5 表示未登录,-2 表示密码错误，-3 表示用户不存在，>=0 表示登录成功，返回用户位置
        int last_hover[2] = {0};//上次悬停状态

        int state=0;//判断是否在登录注册界面

        UserList UL={0};
        ReadAllUser(&UL);

        mouse_off_arrow(&mouse);

        draw_basic();

        mouse_on_arrow(mouse);

        while(1){

            int now_hover[2];
```

```
            now_hover[0] = mouse_location(350, 490, 485, 540); // 是否
悬浮在登录按钮上
            now_hover[1] = mouse_location(515, 490, 650, 540); // 是否
悬浮在注册按钮上

            mouse_show_arrow(&mouse);

            if(state==0)//在登录注册界面
            {
                if(mouse_press(515, 490, 650, 540)==1)//点击注册
                {
                    DestroyUList(&UL);//销毁线性表
                    user_register();//进入注册页面
                    //return 后从这开始

                    ReadAllUser(&UL);//重新读取用户信息
                    mouse_off_arrow(&mouse);
                    draw_basic();
                    mouse_on_arrow(mouse);
                }
                else if(mouse_press(750,50,850,100)==1)
                {
                    draw_about_us();//关于我们
                    state=1;
                }
                else if(mouse_press(900,50,950,100)==1)
                {
                    draw_about_product();//关于产品
                    state=1;
                }
                 else if(mouse_press(350, 330, 650, 380)==1)//点击账号
框
                {
```

```
                    input_mode(name,code,judge, 355, 335, 645,
375,1,1);//输入账号
                }
                 else if(mouse_press(350, 410, 650, 460)==1)//点击密码
框
                {
                    input_mode(name,code,judge, 355, 415, 645,
455,2,1);//输入密码
                }
                else if(mouse_press(350, 490, 485, 540)==1)//点击登录
                {
                     result=Check_info(UL,name,code);//判断账号密码是否
正确

                   if(result >= 0)
                   {
                        int user_type = UL.elem[result].type; // 获取
用户类型
                       users.pos=result  ;//记录用户位置

                       DestroyUList(&UL);//销毁线性表

                       // 根据用户类型跳转到不同界面，并传入用户位置
                       if (user_type == 1)
                       {
                           user(users.pos); // 用户页面
                       }
                       else if (user_type == 2)
                       {
                           business(users.pos); // 商家页面
                       }
                       else if (user_type == 3)
                       {
```

```
                        rider(users.pos); // 骑手页面
                    }
                    //return 后从这开始
                    ReadAllUser(&UL);
                    mouse_off_arrow(&mouse);
                    draw_basic();
                    mouse_on_arrow(mouse);

                    // 清空输入框内容
                     for (i = 0; i < sizeof(name); i++) name[i] =
'\0';
                     for (i = 0; i < sizeof(code); i++) code[i] =
'\0';
                    for (i = 0; i < sizeof(judge); i++) judge[i] =
'\0';
                }
                if(result ==-2)//密码输入错误
                {
                    PrintCC(430,550,"密码错误",HEI,24,1,lightred);
                    delay(500);
                    bar1(430,550,580,580,white);
                }
                if(result ==-3)//用户不存在
                {
                                PrintCC(430,550," 用 户 不 存 在
",HEI,24,1,lightred);
                    delay(500);
                    bar1(430,550,580,580,white);
                }
            delay(15);
            }

            // 悬停效果
```

```
                if (now_hover[0] != last_hover[0])
                {
                    if (now_hover[0] == 1)
                    {
                        mouse_off_arrow(&mouse);
                         Fill_Rounded_Rectangle(350, 490, 485, 540, 5,
white);
                         Draw_Rounded_Rectangle(350, 490, 485, 540, 5,
1, lightblue);
                              PrintCC(390, 503, "登录", HEI, 24, 1,
lightblue);
                        mouse_on_arrow(mouse);
                    }
                    else
                    {
                        mouse_off_arrow(&mouse);
                         Fill_Rounded_Rectangle(350, 490, 485, 540, 5,
lightblue);
                        PrintCC(390, 503, "登录", HEI, 24, 1, white);
                        mouse_on_arrow(mouse);
                    }

                    last_hover[0] = now_hover[0]; // 更新状态
                }
                if (now_hover[1]!= last_hover[1])
                {
                    if (now_hover[1] == 1)
                    {
                        mouse_off_arrow(&mouse);
                         Fill_Rounded_Rectangle(515, 490, 650, 540, 5,
white);
                         Draw_Rounded_Rectangle(515, 490, 650, 540, 5,
1, lightblue);
```

```
                            PrintCC(555, 503, "注册", HEI, 24, 1,
lightblue);
                        mouse_on_arrow(mouse);
                    }
                    else
                    {
                        mouse_off_arrow(&mouse);
                         Fill_Rounded_Rectangle(515, 490, 650, 540, 5,
lightblue);
                        PrintCC(555, 503, "注册", HEI, 24, 1, white);
                        mouse_on_arrow(mouse);
                    }

                    last_hover[1] = now_hover[1]; // 更新状态
                }

            }

            if(state==1)
            {
                if(mouse_press(750,50,850,100)==1)
                {
                    draw_about_us();//关于我们
                    state=1;
                }
                else if(mouse_press(900,50,950,100)==1)
                {
                    draw_about_product();//关于产品
                    state=1;
                }
                else if(mouse_press(700, 225,720,245)==1)
                {
                    mouse_off_arrow(&mouse);
```

```
                draw_basic();//返回登录界面
                mouse_on_arrow(mouse);
                state=0;
            }
        }
    }
}

void draw_basic()
{
    Readbmp64k(0, 0, "bmp\\city.bmp");

    Fill_Rounded_Rectangle(310, 200, 690, 580, 25,white);//填色

    Fill_Rounded_Rectangle(350, 330, 650, 380, 5,lightgrew);//账号栏填色
    Fill_Rounded_Rectangle(350, 410, 650, 460, 5,lightgrew);//密码栏填色

    Fill_Rounded_Rectangle(350, 490, 485, 540, 5,lightblue);//登录按钮//长 185，宽 50
    Fill_Rounded_Rectangle(515, 490, 650, 540, 5,lightblue);//注册按钮//圆角方框，两字，x65，y+13

    PrintCC(355,245,"校园外卖快递平台",HEI,32,1,0);
    PrintCC(355,285,"基于华中科技大学校园制作",HEI,16,1,0XC618);
    PrintCC(355,345,"账号",HEI,24,1,deepgrew);
    PrintCC(355,425,"密码",HEI,24,1,deepgrew);
    PrintCC(390,503,"登录",HEI,24,1,white);
    PrintCC(555,503,"注册",HEI,24,1,white);
    PrintCC(750,50,"关于我们",HEI,16,1,white);
    PrintCC(900,50,"关于产品",HEI,16,1,white);
```

```
    }

    void draw_about_us()
    {
        Fill_Rounded_Rectangle(250, 200, 750, 580, 30, white); // 背景
框

        Line_Thick(700, 225, 720, 245, 1, black);//"x"
        Line_Thick(700, 245, 720, 225, 1, black);//

        PrintText(270, 225, "联系我们", HEI, 32, 1, 0x0000); // 标题
（黑色）

        PrintText(270, 300, "本系统由华中科技大学人工智能与自动化", HEI,
24, 1, 0x0000);
        PrintText(270, 330, "学院智能 2402 班的曹瀚鹏，张子恒完成。", HEI,
24, 1, 0x0000);
        PrintText(270, 360, "曹瀚鹏主要负责完成骑手端编写及路径", HEI,
24, 1, 0x0000);
        PrintText(270, 390, "规划操作，张子恒主要负责完成用户和", HEI, 24,
1, 0x0000);
        PrintText(270, 420, "商家信息存储，订单展示功能", HEI, 24, 1,
0x0000);
        PrintText(270, 450, "如有问题，请联系：", HEI, 24, 1, 0x0000);
        PrintText(270, 480, "曹瀚鹏  2960473693", HEI, 24, 1, 0x0000);
        PrintText(270, 510, "张子恒  2496100220", HEI, 24, 1, 0x0000);
    }
    void draw_about_product()
    {
        Fill_Rounded_Rectangle(250, 200, 750, 580, 30, white); // 背景
框

        Line_Thick(700, 225, 720, 245, 1, black);//"X"
```

```
        Line_Thick(700, 245, 720, 225, 1, black);//

        PrintText(270, 225, "产品介绍", HEI, 32, 1, 0x0000); // 标题
（黑色）

        PrintText(270, 300, "本系统模拟了校园内外卖配送超市购物配", HEI,
24, 1, 0x0000);
        PrintText(270, 330, "送及快递代取的完整流程，涵盖用户注册", HEI,
24, 1, 0x0000);
        PrintText(270, 360, "与管理、商家管理、订单处理、骑手配送", HEI,
24, 1, 0x0000);
        PrintText(270, 390, "等功能。通过合理的数据选取与系统设计", HEI,
24, 1, 0x0000);
        PrintText(270, 420, "该系统可为校园内的学生、教师、食堂及", HEI,
24, 1, 0x0000);
        PrintText(270, 450, "超市商家、配送骑手等提供一体化的线上", HEI,
24, 1, 0x0000);
        PrintText(270, 480, "交易与配送服务。结合骑手路径优化功能", HEI,
24, 1, 0x0000);
        PrintText(270, 510, "该系统能够提高配送效率，减少配送成本", HEI,
24, 1, 0x0000);
        PrintText(270, 540, "保障订单的及时送达。", HEI, 24, 1, 0x0000);
    }

/*****************************************************************/
```

九、课设感想

1. 张子恒：

这两个月写 c 语言课程设计让我学会了很多，也是我第一次参与编写先项目。一个项目从零到实现，的确要花费很多心血。项目的成功也不仅仅取决于个人的能力，更看重的是团队合作。从一开始选择题目，开题报告，到后面的具体编程过程，与队友的分工合作，代码进度的管理。每一步遇到了很多困难，但最终都一一克服了。在写代码时，上学期学习的知识都逐渐运用到了实

战中，同时也学习到了许多新知识。我负责的部分是文件读写操作，进行各种信息的存储和展示。由于使用的是二进制文件，无法查看，所以在最初写入时遇到很大困难，不知道是否成功写入，后来编写了一段测试代码，成功把文件中的数据在终端展示出来，但也出现了许多乱码，最后通过仔细地对字节进行检查，能够完整地在图形化界面显示信息，再后来完成了不同用户端实现信息同步，把用户下的订单同步到商家端和骑手端，骑手端再通过筛选可以展示出所接的订单，当前订单，历史订单。在调试信息的过程中，没有做好内存管理，导致程序无法稳定运行，多次点击页面便会卡死，后来通过细致的检查，逐行代码注释调试，最终才解决内存问题。此外，我觉得此次写 c 课设还有一个收获是学会了使用 git 进行版本管理，并且和队友使用 github 合并代码。虽然初次使用不太熟练，我和队友也有好几次因为错误操作导致代码被覆盖，最后也还是成功克服了困难。

经过此次 c 课设，我学到了很多，包括但不仅限于代码编写，项目管理，版本管理，团队合作，信息处理等，受益匪浅。最后，感谢曾经帮助过我们的同学和各位老师，感谢你们的帮助，让我们的项目圆满完成。

2. 曹瀚鹏：

这段 C 语言课程设计让我有了十足的进步，是目前参与过的规模最大的项目。从最开始的定选题，设计功能，分工合作，到编写程序时从零开始学的图形库的使用，文件读写操作，dijkstra 算法和贪心策略，遇到了很多困难，但却也都一一克服了。在地图的选取中，最开始并未找到适合我们程序的华科地图，便决定手绘地图，后面才在企业微信上找到了合适的资源。为了骑手端的路线规划功能尽可能完善，接近可使用标准，我们对地图数据也尽量毫无保留获取，没有进行简化压缩。先是在平板上对地图结点逐一编号，而后用测量工具测量道路实际距离，再依次定位每个节点的像素坐标，每一项都是一个繁杂费力的过程，地图数据的获取也都由我花费将近一周完成，力求准确无误。在代码的调试过程中，我们起初在内存管理上存在诸多漏洞，导致程序总是不明原因崩溃卡死，最终，在我和队友的对每行代码逐一注释，添加调试信息，找出了问题的根源所在。在团队代码协作上，首次使用 GitHub 仓库进行多人合作，由于不熟悉英文的操作提示，导致多次版本错乱，写好完成的代码被强制读写覆盖，但我们还是克服了这些困难，掌握了使用 git 管理代码的基本操作，为以后团队开发打下了坚实基础。在此过程中，不仅有我的代码能力，编程能力，信息处理能力都得到了显著提高，团队协作，心态调整的能力也得到了锻炼。

总之，在这次 C 语言课程设计中学到了很多，心态也成熟了很多，在不远的将

来，当我参与更多更庞大更复杂项目时，或许也不会忘记这次 C 语言课程设计为我筑起的稳定的地基吧。

十、课设分工

	张子恒		曹瀚鹏		借用			
1	main.c	14	acp_det.c	292	SVGA.c	778		
2	welcome.c	199	acp_or.c	381	HZK.c	297		
3	user.c	376	arrange.c	205	mouse.c	408		
4	u_re.c	213	button.c	141				
5	u_shop.c	428	lgfunc.c	335				
6	u_food.c	198	m_ac_de.c	266				
7	u_take.c	90	m_acp.c	312				
8	u_order.c	422	m_hst.c	463				
9	u_deliv.c	512	m_inf.c	156				
10	u_cart.c	181	rider.c	243				
11	busi.c	349	route.c	606				
12	bu_order.c	121	timee.c	8				
13	bu_det.c	199	shape.c	133			总行数	11159
14	de_order.c	76					注释行	1094
15	f_order.c	366					空行	1297
总和		3744(51%)		3541(49%)		1483	代码行	8768