# 一、全局变量定义

概要设计报告

int ac = 0;//初始化时间加速倍数 int shine;//初始化订单显示时间

int w1 ;// 未完成订单数最大值 int w2;// 未完成订单数最小值 int time = 0;//初始化时间为 0

int ordernum = 0;//初始化当前未完成订单数为 0

int ordercon = 2;//初始化总订单状态（ordernum>w1 为 1， ordernum<=w1 为 2） int porder = -1;//初始化未完成的最早的订单的编号，初值为-1

food \*foods = NULL;//创建一个结构体数组，用来储存食物的信息 int foodsize = 0;//将数组的大小初始化为 0

set \*sets = NULL;//创建一个结构体数组，用来储存套餐的信息 int setsize = 0;//将数组的大小初始化为 0

order orders = [100];//创建一个结构体数组，用来储存订单的信息 int ordersize = 0;//将数组的大小初始化为 0

int TIME = 0;//初始化图形化界面显示的时间为 0 int check;//初始化当前时刻是否点单的标记

char timept[10];//将时间表示成时分秒的形式 ExMessage a;//初始化鼠标结构体信息 IMAGE img;//初始化图片结构体信息

# 二、数据结构的定义

1. 食物的结构：

typedef struct Food

{

char foodname [51]; 食物的名称 int foodtime; 食物制作所需时间 int foodmax; 食物的最大储存容量 int remain; 食物在当前秒的存量

int ordernum; 食物被订单分配的个数

int orderfinnum; 食物被分配已经完成的个数

int timer; 距离上一个食物制作周期的时间

} food;

1. 套餐的结构：

typedef struct Set

{

char setname [51]; 套餐的名称

int foodid [5]; 套餐中所含的食物的代码

int size; 套餐所含食物的种类

} set;

1. 订单的结构：

typedef struct Order

{

int ordertime; 订单下单的时间

int foodid [5]; 订单所含的食物的代码

int foodcon [5]; 订单所含食物的状态（0 表示未完成，1 表示已完成）

int size;订单所含食物的种类 int finishtime; 订单完成的时间

int con; 订单的状态（con）(0 表示未分配，1 表示已分配未完成，2 表示已完

成，3 表示失败)

} order;

1. **小组成员与分工**

某某某40%：负责整体代码框架以及左侧食物、套餐的显示与点餐的逻辑

某某某30%：负责部分核心函数的代码实现，如时间显示功能、显示已完成与未完成订单功能、退出按键功能，以及初版界面布局的构思

某某某30%：负责美化图形化界面，包括设置图片背景、设置文字的格式与显示逻辑，以及使用拖拽进度条实现时间加速功能

# 四、函数的定义

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序 号 | 函数原型 | 功能 | 参数 | 返回值 |
| 1 | int findfood (char a []) | 根据食物名称查找食  物的id | a是食物名称的字符串 | 查找食物的id，并返回食物的id,如果未找到，则返回-1  。 |
| 2 | int findset (char a []) | 根据套餐名称查找套餐的id | a是套餐名称的字符串 | 查找套餐的id，并返回套餐的id，如果未找到，则返回- 1。 |
| 3 | int judgeorder (int a) | 判断给出的订单是否完成 | a是一个订单的id | 若给出的订单达成完成的条件返回2，若未达成则返回1  。 |
| 4 | int checkorder (int a) | 检查目前未完成订单的总数量并判断订单状态，并控制系统开闭。 | a是标记 | 若可以继续接单则返回1，若不能继续接单则返回0 |
| 5 | int findorder () | 查找指定时间中的订单 | void | 返回找到的订单的id，没有相应的订单则返回-1 |
| 6 | void initializefoodset () | 初始化食物和套餐的结构体数组，并设定 w1,w2 | void | void |
| 7 | void makefood () | 进行每秒食品的制作 | void | void |
| 8 | void allocateorder (int a) | 将一个订单分配给食物 | a是一个订单的id | void |
| 9 | void processorder (int a,int c) | 处理当前未完成订单 | a是处理的订单的id，c是该订单是否为当前秒数的订单的标记 | void |
| 10 | void printorder () | 将处理后的订单信息储存在output.txt中 | Void | void |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 11 | void doorder() | 集成一秒内处理订单的程  序 | | void | void | | |
| 12 | void printText(int x, int y, const char\* text, int fontSize, const char\* fontType, COLORREF fillColor, COLORREF  textColor) | | 在指定位置打印文本，不改变字体格式 | | | x坐标，y坐标，字符串，字体大小，字体格式，填充颜色，字体颜色 | void |
| 13 | void drawRoundRect(int left, int top, int right, int bottom, int aa, int bb, COLORREF fillColor) | | 在指定位置打印圆角矩形，不改变填充颜色 | | | Left左侧位置，top顶部位置， right右侧位置，bottom底部位置  ，aa左侧圆角弯曲程度，bb右侧圆角弯曲程度，fillcolor填充  颜色 | void |
| 14 | void drawRoundRectWithBorder(int left, int top, int right, int bottom, int aa, int bb, COLORREF fillColor, COLORREF borderColor) | | 在指定位置打印带边框的圆角矩形，不改变填充颜色 | | | Left左侧位置，top顶部位置， right右侧位置，bottom底部位置  ，aa左侧圆角弯曲程度，bb右侧圆角弯曲程度，fillcolor填充  颜色，bordercolor边框颜色 | void |
| 15 | void drawfoodset() | | 初始化图形点餐界面， | | | Void | Void |
| 16 | void drawinitiate() | | 初始化点餐按钮 | | | void | Void |
| 17 | void failprint() | | 检测下单失败并输出失败的是几号订单 | | | void | void |
| 18 | void printunfinish() | | 在右侧界面正在制作界面输出正在制作的订单 | | | void | void |
| 19 | void printfinish() | | 在右侧界面完成订单界面输出已完成的订单 | | | Void | Void |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 20 | void drawrightside() | 绘制右侧图形化界面 | Void | Void |
| 21 | void print\_the\_time() | 输出时钟时分秒文本 | Void | Void |
| 22 | void drawtheclock() | 绘制时钟 | void | Void |
| 23 | void change\_TIME() | 时间加速函数 | Void | Void |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 24 | void drawexit() | 绘制退出按钮 | void | void |
| 25 | void checkclick() | 检测鼠标点击，进行点餐 | void | void |
| 26 | void clickorderfood(int x) | 进入食物订单确认界面，显示订单详情、订单号及下单时间 | X为食物ID | void |
| 27 | void clickorderset(int x) | 进入套餐订单确认界面，显示订单详情、订单号及下单时间 | X为套餐ID | void |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 28 | void drawProgressBar(ProgressBar\* bar) | 绘制时间加速进度条 | Bar指向 ProgressBar 结构体的指针。这个结构体包含了绘制进度条所需的信息，如进度条的位置 (x, y) 和当前进度值  (value)。 | Void |
| 29 | void dragbar（） | 检测鼠标位置，实现拖动，进而加速 | void | void |
| 30 | void changefin(int a) | 将订单fin时间转换为标准格式 | int型，a为订单ID | void |
| 31 | void changestart(int a) | 将订单start时间转换为标准格式 | int型，a为订单ID | Void |
| 32 | void print() | 将所有订单情况标准输出 | Void | Void |

**五、用户界面设计**

本程序用户界面设计如图所示。（图片附在该文档最后）

（1）系统界面概览

开始界面：提供了简洁的欢迎信息，并设有“点餐”和“Exit”（退出）按钮，供用户选择进入点餐流程或退出系统。

处理界面：显示了食品单品名称、余量、制作进程以及套餐名称和余量。此外，还有“完成订单”和“当前状态”的显示区域，以及“speed up”（加速）按钮，

用于加快订单处理速度，同时支持显示系统关闭下单失败的信息。

点餐界面：与处理界面类似，但增加了“点餐时间”和“订单号”的显示，以及“确定”和“取消”按钮，供用户在点餐时进行确认或取消操作。

（2）功能特点

实时更新：系统能够实时显示食品与套餐的余量和制作状态，确保顾客了解当前可点餐的食品情况。

订单管理：用户可以通过系统查看订单号、预计完成时间与订单状态，便于追踪订单状态。

操作简便：界面设计简洁直观，用户易于上手，通过点击相应按钮即可完成点餐或退出操作。

加速功能：系统提供加速按钮，在点击加速按钮后可拖动进度条，在1~100倍中任意选择加速倍数，再次点击按钮确认，可用于加快订单处理速度。

输出结果：在退出系统后，会自动生成同oj版一样的标准输出，同时会显示所有订单的编号、所含食物、下单时间、完成时间以及最后状态

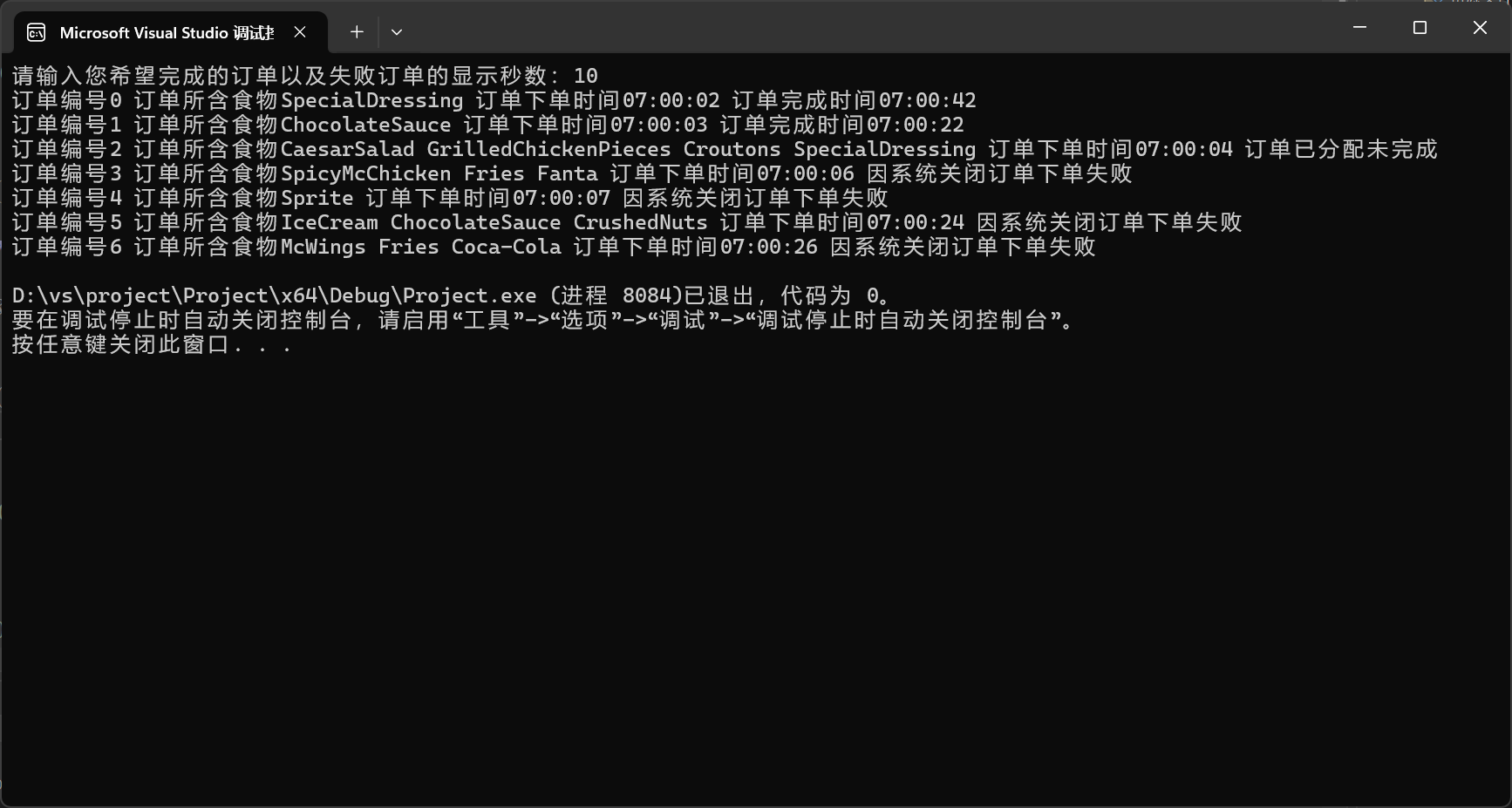
（3）界面细节

食物单品与套餐：界面列出了包括BigMac、Fries、McChicken等在内的多种食品单品和套餐选项。

余量与制作进程：每种食品旁边都显示了余量和预计制作完成时间，如BigMac的余量为0，预计15秒后完成制作。

订单信息：界面上显示了订单号和完成时间，用户可以根据这些信息了解自己订单的进度。

**处理界面**

****

输出页面

点餐页面

开始界面