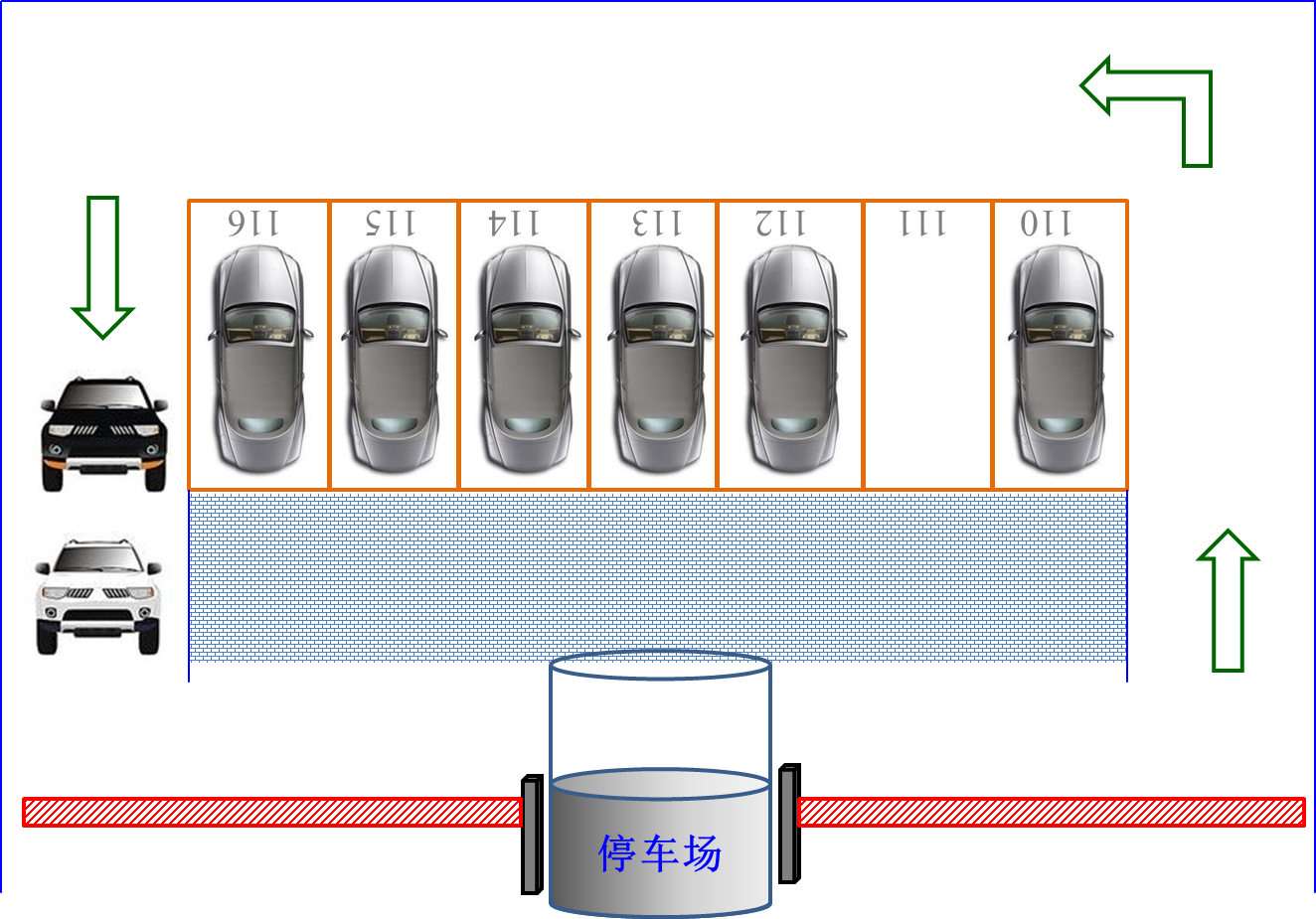
**模拟停车场管理系统**

1. **实验目的**

使用C/C++语言实现模拟停车场管理系统，并符合基本要求。

1. **实验内容**

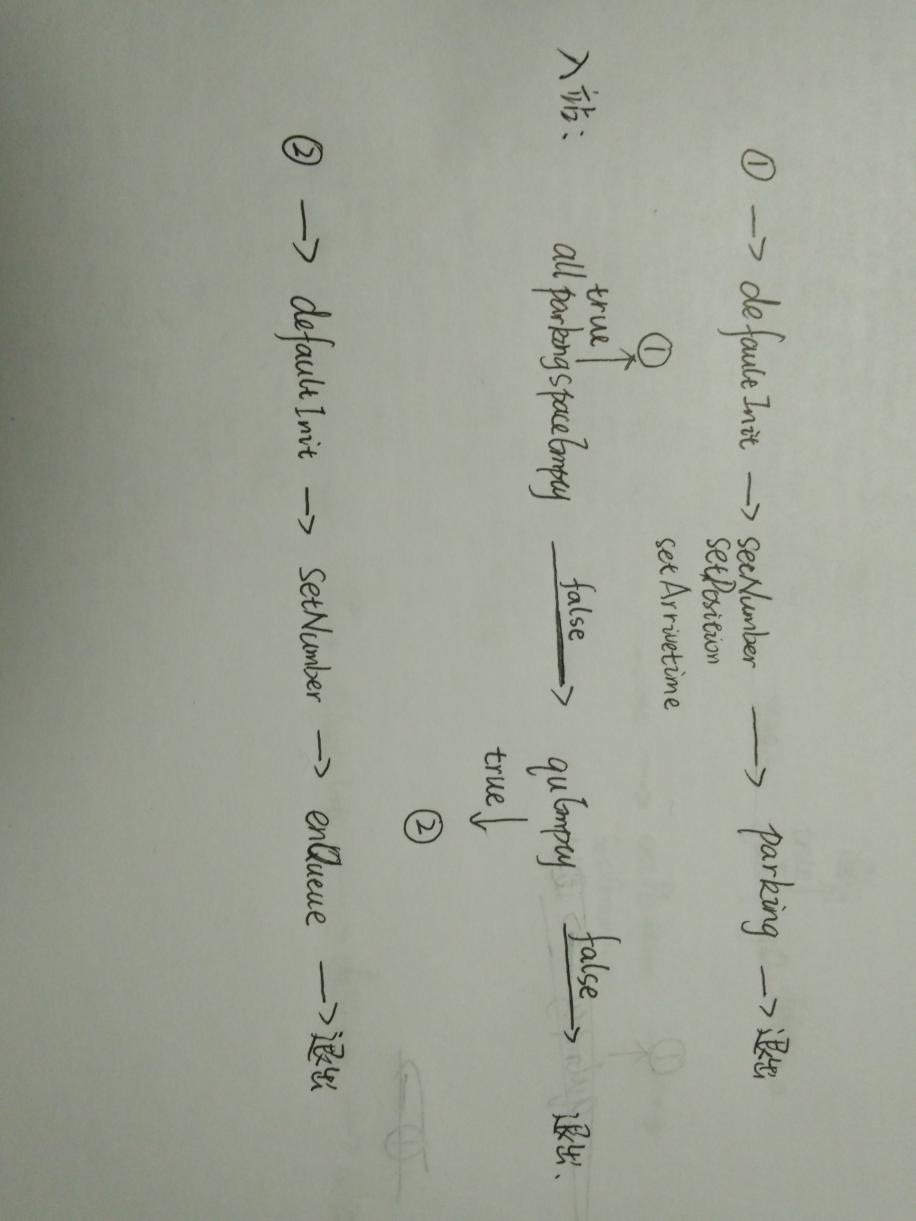
**模拟停车场如下图所示**

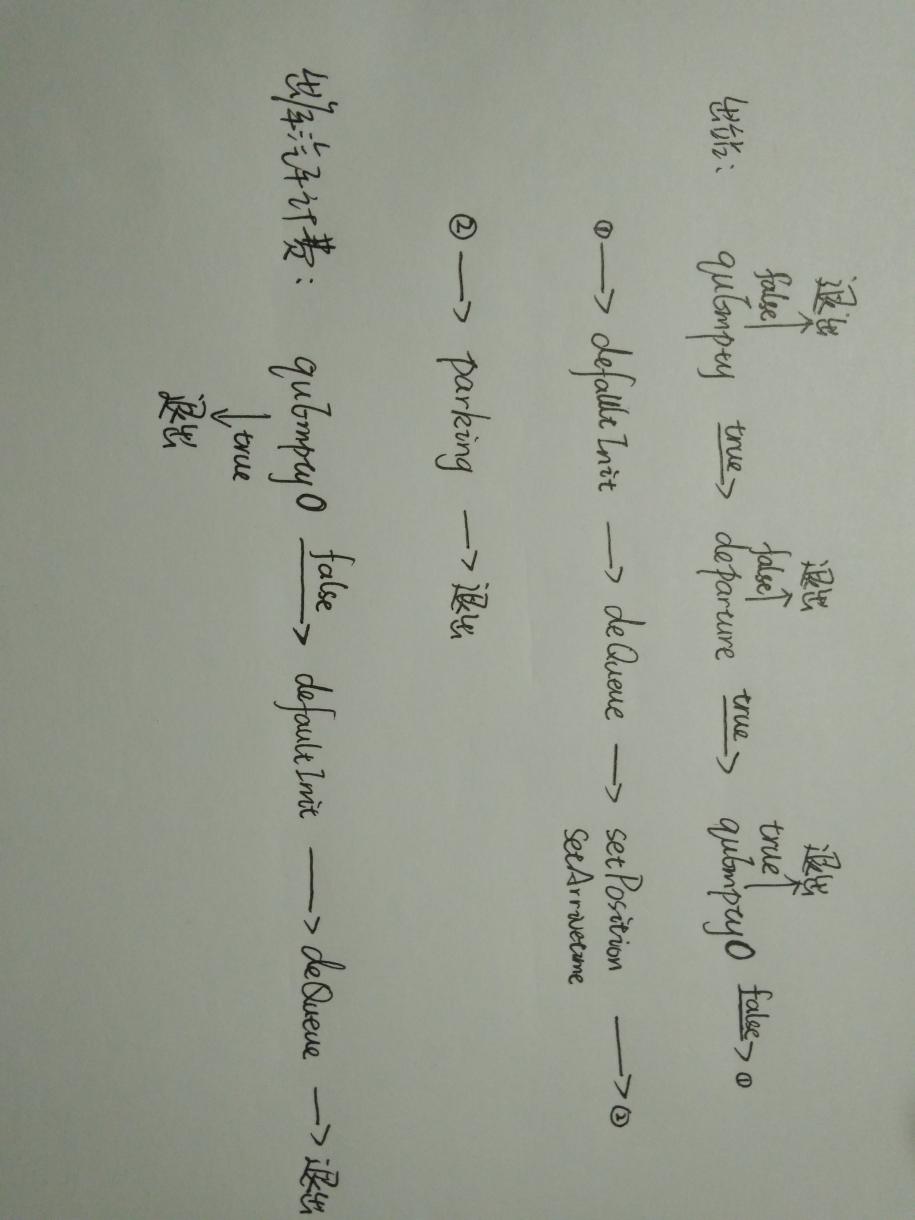
****

停车场是可停放n辆汽车的狭长通道，停车场内只有一条单行线通道可走车，有一个大门可供汽车进出。

**基本要求：**

1. 进站：若车场内已停满n辆汽车，则后来的汽车只能在门外通道上**等候**，一旦有车开走，则排在通道上的第一辆车即可开入。
2. 出站：当停车场内某辆汽车要离开时，必须按它在停车场停留的时间长短缴纳费用；如果通道前面有汽车在**缴纳费用**，则要在其后面排队**等候**。
3. 假设出、入便道上一次只能排m辆汽车，本题按n=7，m=3进行模拟，时间计算到分钟，每小时停车费3元。
4. 假设**初始**时停车场内停了6辆汽车，没有排队等候进站或出站的汽车。
5. **项目程序结构（程序中的函数调用关系图）**





1. **概要设计**
   1. **数据类型定义（含所用到的结构的表示及其基本操作）**

（1）汽车数据类型Data

typedef struct{

int position; //停车位置

int number; //车牌号码

int arrive\_hour; //到达时间\_小时

int arrive\_minute; //到达时间\_分钟

}Data;

函数：

①setPosition(Data &D,int position)//输入停车位置

②setNumber(Data &D,int number)//输入车牌号码

③setArrivetime(Data &D,int arrive\_hour,int arrive\_minute)//设置到达时间

④defaultInit(Data &D)//默认设置

1. 排队队列类型SqQueue（用到循环队列数据结构类型）

typedef Data ElemType;

typedef struct Queue{

ElemType elem[m];

int front;

int rear;

}enter,departrue;

//队列类型\_入站队列，出站队列

函数：

①InitQueue(Queue &Q)//初始化队列

②enQueue(Queue &Q,ElemType e)//进入队列

③deQueue(Queue &Q,ElemType &e)//出队列，并赋值给e

④quEmpty(Queue &Q)//判断队列是否为有空位

⑤quEmpty0(Queue &Q)//判断队列是否为空

⑥DisplayQueue\_en(Queue Q)//输出入站等候队列

⑦DisplayQueue\_de(Queue Q)//输出出站等候队列

1. 车位类型SqList（用到顺序表数据结构类型）

typedef Data ElemType;

typedef struct{

ElemType elem[n];

}SqList;

//车位信息

函数：

①parkingspaceEmpty(ElemType e)//某个车位是否为空

②allparkingspaceEmpty(SqList L)//判断是否有空车位

③setDefault(SqList &L,int position)//车位信息默认设置(车辆离开时)

④setAllDefault(SqList &L)//初始化设置

⑤parking(SqList &L,ElemType e,int position)//入站停车

⑥departure(SqList &L,Queue &dep,int number)//离开车位进入等候出站队列

⑦Display(SqList L)//显示当前车位信息

* 1. **主函数main设计**

联用while-switch语句块构成可循环的程序，具体功能实现如下：

①入停车场：

输入信息存入Data数据模型中，然后调用parking将该Data数据模型存入SqList模型中；

②入进站队列：

输入信息存入Data数据模型中，然后调用enQueue将该Data数据模型存入SqQueue模型中；

③入出站队列：

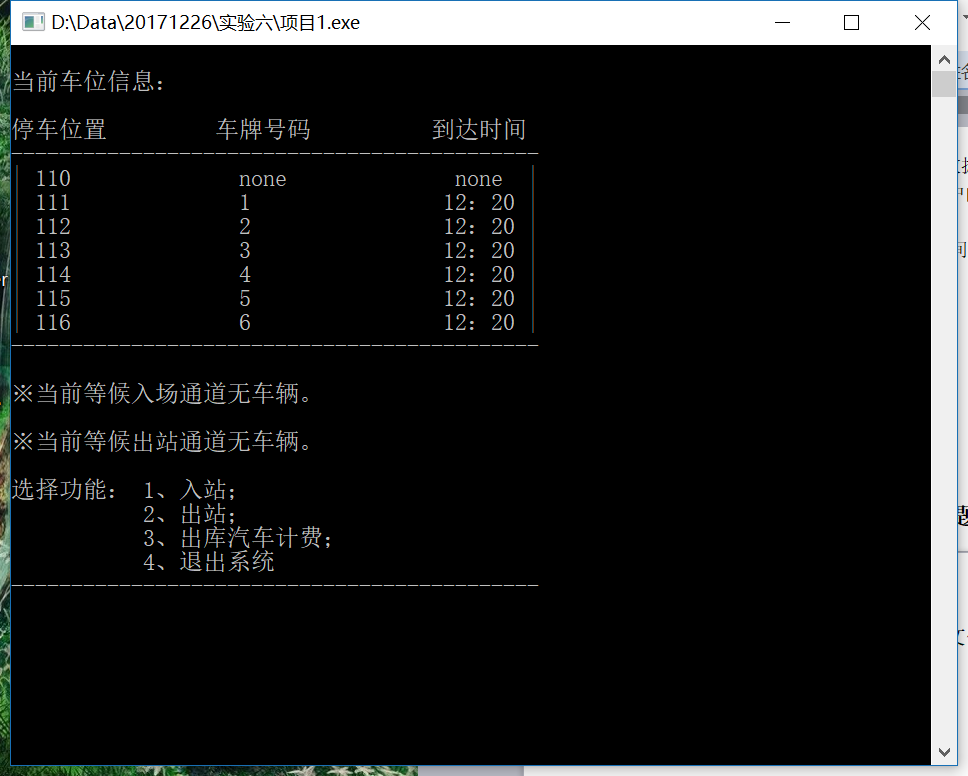
调用enQueue将Data数据模型存入SqQueue，其中Data数据模型来源是将从用户获取的车位位置获取SqList中的Data数据模型；

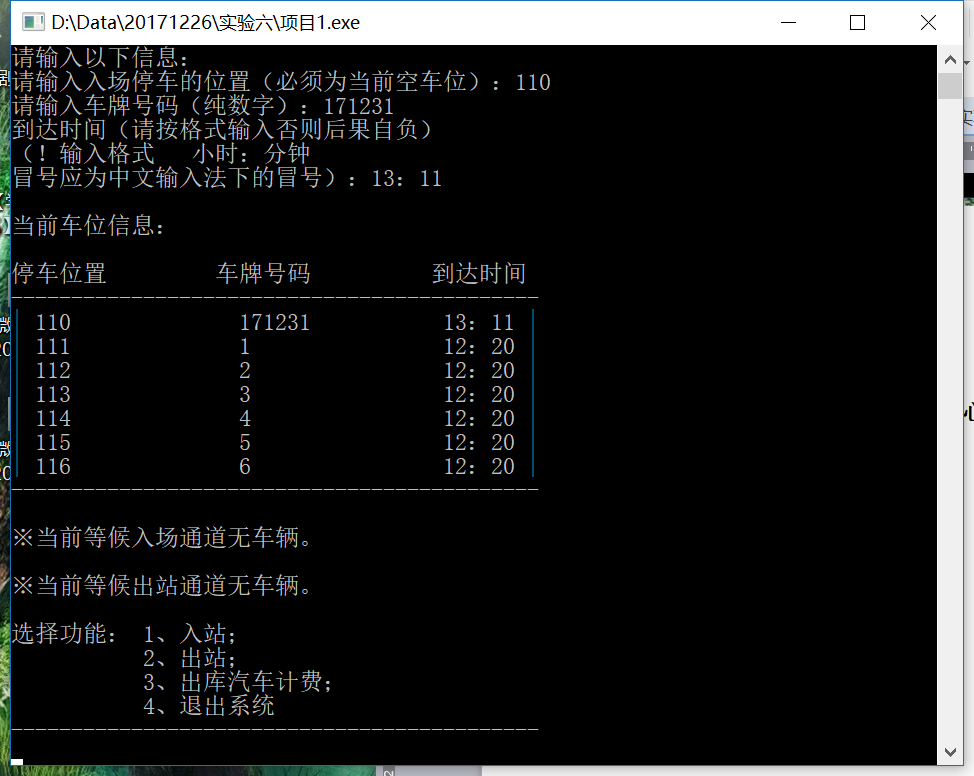
④计费：

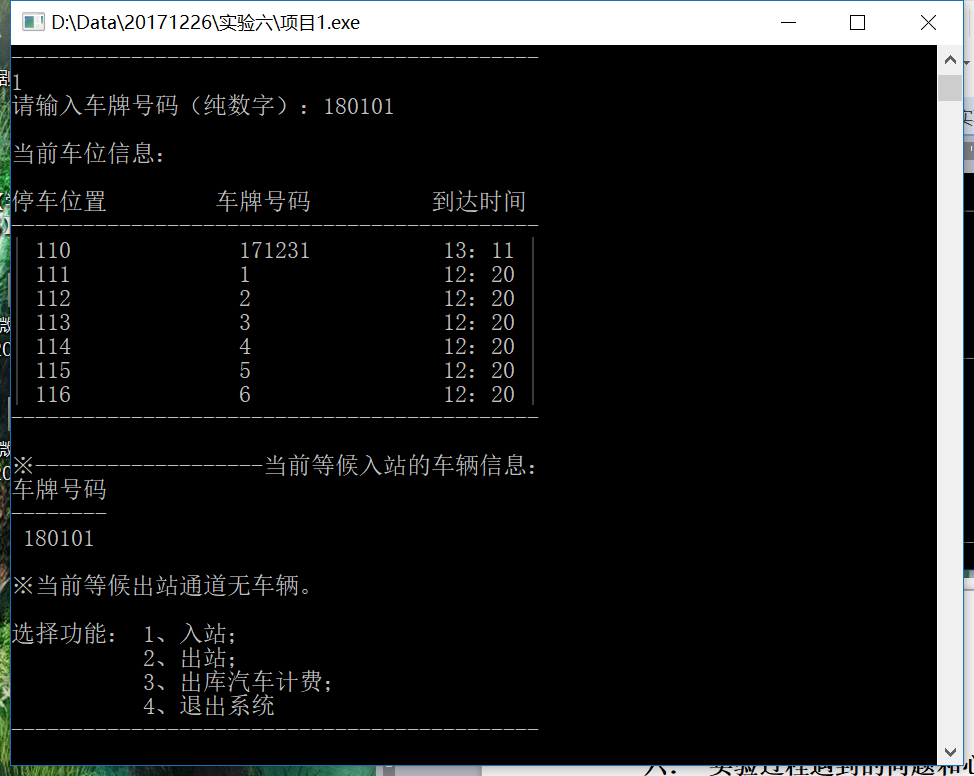
从用户输入获取当前时间，然后计算分钟再计算小时（浮点类型）乘以每小时费用。

1. **测试结果**

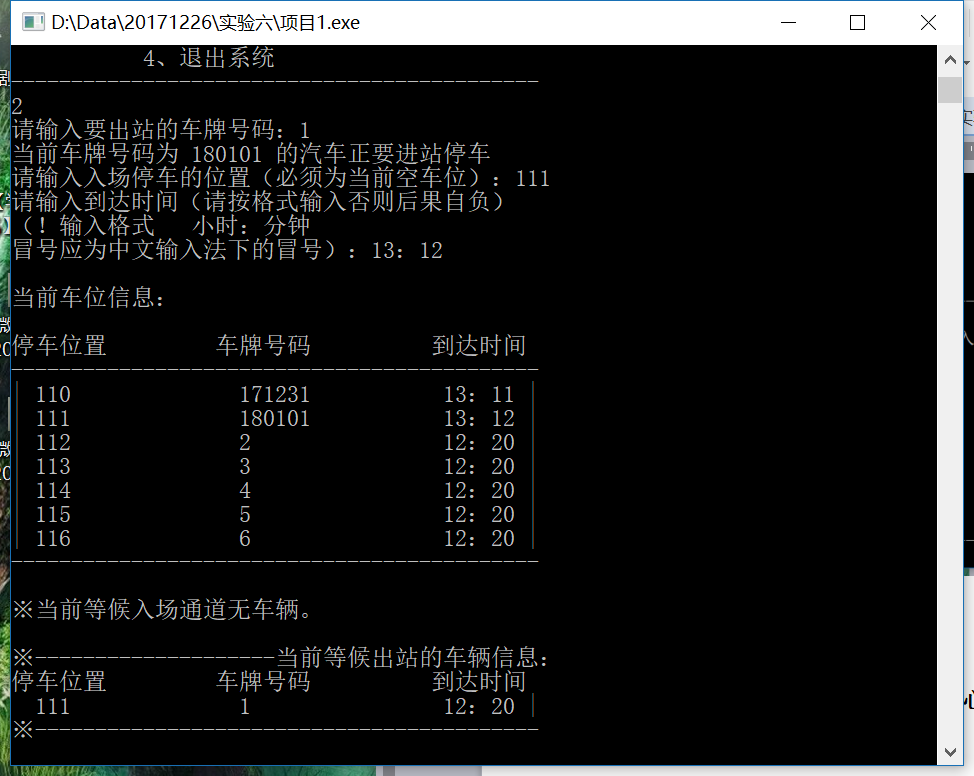
①初始：

②入站（两辆）：

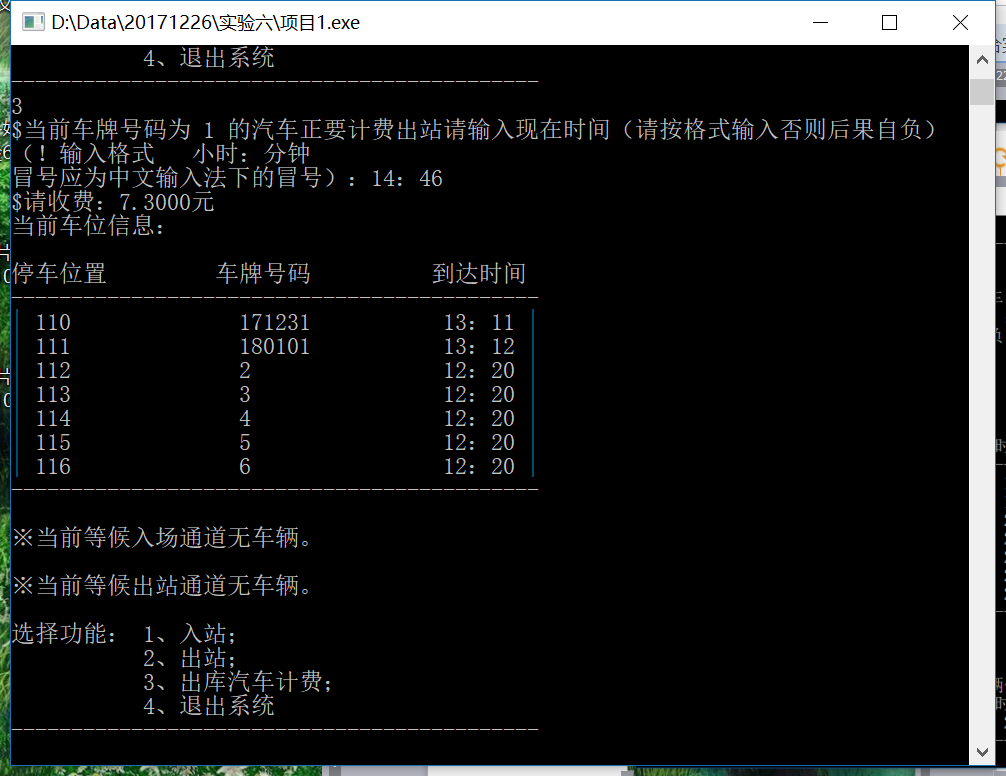




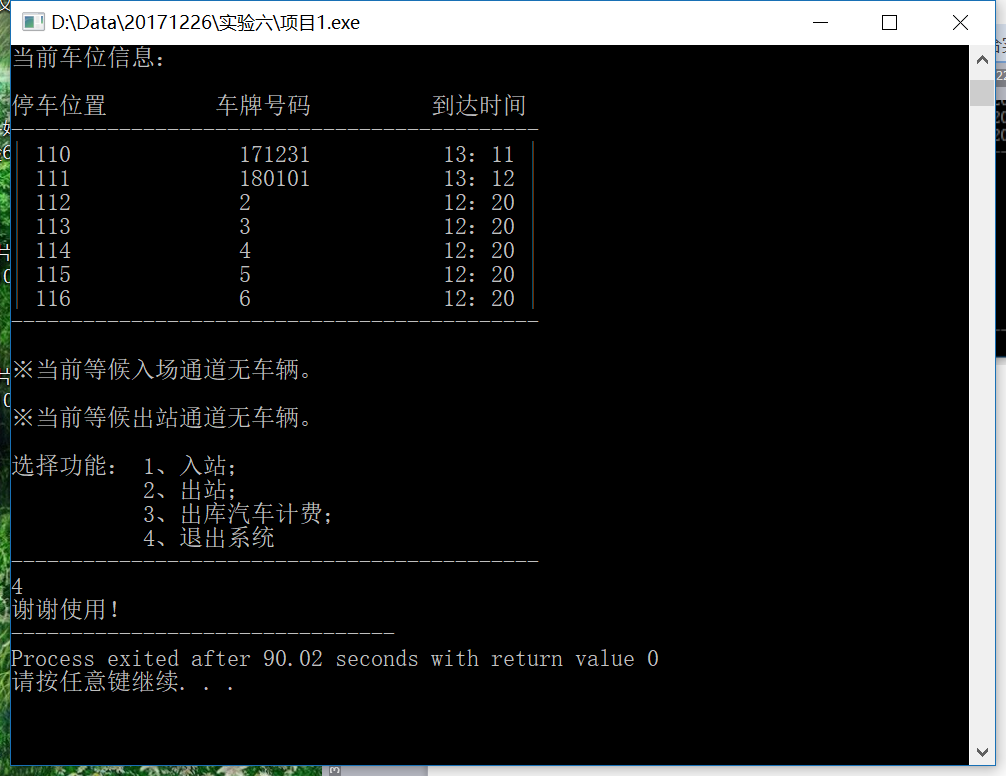
③出站：



④出库汽车计费：



⑤退出系统：



1. **实验过程遇到的问题和心得**

①关于头文件：一个自定义的头文件不可以多次被引用（例如再SqQueue.h中引用Data.h，再在main.cpp中引用Data.h）；

②关于各种数据输入的问题和输出格式的问题，要经过多次调整；

③计算出库汽车费用的时候，如果要用到浮点型不能直接使用，应先扩大适当倍数，然后输出时再缩小同样倍数，否则输出为0；

④关于是否有空车位和队列是否为空的问题：这是不同的问题！

⑤要建立一个能够标识的初始化设置判断一个车位是否为空，并且可以初始化Data数据模型的。

附：完整源程序（含可执行文件）

文件：Data.h

typedef struct{

int position; //停车位置

int number; //车牌号码

int arrive\_hour; //到达时间\_小时

int arrive\_minute; //到达时间\_分钟

}Data;

//定义车的数据模型

bool setPosition(Data &D,int position){

if(position>=110&&position<=116){

D.position=position;

return true;

}

else{

return false;

}

}

//输入停车位置

void setNumber(Data &D,int number){

D.number=number;

}

//输入车牌号码

bool setArrivetime(Data &D,int arrive\_hour,int arrive\_minute){

if(arrive\_hour>=0&&arrive\_hour<24&&arrive\_minute>=0&&arrive\_minute<60){

D.arrive\_hour=arrive\_hour;

D.arrive\_minute=arrive\_minute;

return true;

}

else {

return false;

}

}

//设置到达时间

void defaultInit(Data &D){

D.position=0;

D.number=-1;

D.arrive\_hour=0;

D.arrive\_minute=0;

}

//默认设置

文件SqQueue.h

#define m 4

#include "Data.h"

typedef Data ElemType;

typedef struct Queue{

ElemType elem[m];

int front;

int rear;

}enter,departrue;

//队列类型\_入站队列，出站队列

void InitQueue(Queue &Q){

Q.front=0;

Q.rear=0;

}

//初始化队列

bool enQueue(Queue &Q,ElemType e){

if((Q.rear+1)%m==Q.front){

return false;

}

else{

Q.rear=(Q.rear+1)%m;

Q.elem[Q.rear]=e;

return true;

}

}

//进入队列

bool deQueue(Queue &Q,ElemType &e){

if(Q.front==Q.rear){

return false;

}

else{

Q.front=(Q.front+1)%m;

e=Q.elem[Q.front];

return true;

}

}

//出队列，并赋值给e

bool quEmpty(Queue &Q){

if((Q.rear+1)%m!=Q.front){

return true;

}

else{

return false;

}

}

//判断队列是否为有空位

bool quEmpty0(Queue &Q){

if(Q.front==Q.rear){

return true;

}

else{

return false;

}

}

//判断队列是否为空

void DisplayQueue\_en(Queue Q){

if(quEmpty0(Q)){

printf("\n※当前等候入场通道无车辆。\n");

}

else{

printf("\n※-------------------当前等候入站的车辆信息：\n");

printf("车牌号码\n");

printf("--------\n");

for(int i=(Q.front+1)%m;i!=(Q.rear+1)%m;i=(i+1)%m){

printf(" %d",Q.elem[i].number);

printf("\n");

}

}

}

//输出入站等候队列

void DisplayQueue\_de(Queue Q){

if(quEmpty0(Q)){

printf("\n※当前等候出站通道无车辆。\n");

}

else{

printf("\n※--------------------当前等候出站的车辆信息：\n");

printf("停车位置 车牌号码 到达时间\n");

for(int i=(Q.front+1)%m;i!=(Q.rear+1)%m;i=(i+1)%m){

printf(" %d ",Q.elem[i].position);

printf("%d",Q.elem[i].number);

int q,k=1;

for(q=1;q<=16;q++){

k=k\*10;

if(Q.elem[i].number/k==0){

break;

}

}

for(int l=0;l<=16-q;l++){

printf(" ");

}

if(Q.elem[i].arrive\_hour<=9){

printf("0%d：",Q.elem[i].arrive\_hour);

}

else{

printf("%d：",Q.elem[i].arrive\_hour);

}

if(Q.elem[i].arrive\_minute<=9){

printf("0%d |\n",Q.elem[i].arrive\_minute);

}

else{

printf("%d |\n",Q.elem[i].arrive\_minute);

}

}

printf("※------------------------------------------\n");

}

}

//输出出站等候队列

文件SqList.h

#include <stdio.h>

#include "SqQueue.h"

#define n 7

typedef Data ElemType;

typedef struct{

ElemType elem[n];

}SqList;

//车位信息

bool parkingspaceEmpty(ElemType e){

if(e.number==-1&&e.arrive\_hour==0&&e.arrive\_minute==0){

return true;

}

else{

return false;

}

}

//某个车位是否为空

bool allparkingspaceEmpty(SqList L){

bool empty=false;

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

if(parkingspaceEmpty(L.elem[i])){

empty=true;

break;

}

}

return empty;

}

//判断是否有空车位

bool setDefault(SqList &L,int position){

if(position>=110&&position<=116){

L.elem[position-110].position=position;

L.elem[position-110].number=-1;

L.elem[position-110].arrive\_hour=0;

L.elem[position-110].arrive\_minute=0;

return true;

}

else{

return false;

}

}

//车位信息默认设置(车辆离开时)

void setAllDefault(SqList &L){

for(int i=110;i<117;i++){

setDefault(L,i);

}

for(int i=1;i<7;i++){

L.elem[i].number=i;

L.elem[i].arrive\_hour=12;

L.elem[i].arrive\_minute=20;

}//自定义条件

}

//初始化设置

bool parking(SqList &L,ElemType e,int position){

if(position>=110&&position<=116&&parkingspaceEmpty(L.elem[position-110])){

L.elem[position-110]=e;

return true;

}

else{

return false;

}

}

//入站停车

bool departure(SqList &L,Queue &dep,int number){

bool a=false;

if(quEmpty(dep)){

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

if(L.elem[i].number==number){

a=enQueue(dep,L.elem[i]);

if(a){

setDefault(L,L.elem[i].position);

}

break;

}

}

return a;

}

else{

return false;

}

}

//离开车位进入等候出站队列

void Display(SqList L){

printf("\n停车位置 车牌号码 到达时间\n");

printf("--------------------------------------------\n");

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

printf("| %d ",i+110);

if(parkingspaceEmpty(L.elem[i])){

printf("none none |\n");

}

else{

printf("%d",L.elem[i].number);

int q,k=1;

for(q=1;q<=16;q++){

k=k\*10;

if(L.elem[i].number/k==0){

break;

}

}

for(int l=0;l<=16-q;l++){

printf(" ");

}

if(L.elem[i].arrive\_hour<=9){

printf("0%d：",L.elem[i].arrive\_hour);

}

else{

printf("%d：",L.elem[i].arrive\_hour);

}

if(L.elem[i].arrive\_minute<=9){

printf("0%d |\n",L.elem[i].arrive\_minute);

}

else{

printf("%d |\n",L.elem[i].arrive\_minute);

}

}

}

printf("--------------------------------------------\n");

}

//显示当前车位信息

文件main.cpp

#include <iostream>

#include <String.h>

#include "SqList.h"

#define B 3//每小时三元

/\* run this program using the console pauser or add your own getch, system("pause") or input loop \*/

int main(int argc, char\*\* argv) {

int choose; //功能序号

bool run=true; //程序运行状态

bool data\_position=false; //输入位置合法状态

bool data\_time=false; //输入时间合法状态

Data car; //车

int position; //停车位置

int number; //车牌号码

int arrive\_hour; //到达时间\_小时

int arrive\_minute; //到达时间\_分钟

int hour; //现在时间\_小时

int minute; //现在时间\_分钟

float sum; //收费

SqList parkingspace; //车位

enter waitingqueue\_en; //入站等候队列

departrue waitingqueue\_de; //出站等候队列

InitQueue(waitingqueue\_en);

InitQueue(waitingqueue\_de);

int q=17;

int k=1;

//定义变量

setAllDefault(parkingspace);

while(run){

printf("\n当前车位信息：\n");

Display(parkingspace);

DisplayQueue\_en(waitingqueue\_en);

DisplayQueue\_de(waitingqueue\_de);

printf("\n选择功能： 1、入站；\n 2、出站；\n 3、出库汽车计费；\n 4、退出系统\n--------------------------------------------\n");

scanf("%d",&choose);

switch(choose){

case 1:

if(allparkingspaceEmpty(parkingspace)){

while(!data\_position||!data\_time){

defaultInit(car);

printf("请输入以下信息：\n");

printf("请输入入场停车的位置（必须为当前空车位）：");

scanf("%d",&position);

printf("请输入车牌号码（纯数字）：");

scanf("%d",&number);

while(q>11){

for(q=1;q<=17;q++){

k=k\*10;

if(number/k==0){

break;

}

}

if(q>11){

printf("！输入数字超过11位！\n");

printf("请输入车牌号码（纯数字）：");

scanf("%d",&number);

}

}

q=17;

k=1;

setNumber(car,number);

printf("到达时间（请按格式输入否则后果自负）\n（！输入格式 小时：分钟 \n冒号应为中文输入法下的冒号）：");

scanf("%d：%d",&arrive\_hour,&arrive\_minute);

data\_position=setPosition(car,position);

data\_time=setArrivetime(car,arrive\_hour,arrive\_minute);

if(!data\_position||!data\_time){

printf("输入数据有误请仔细检查输入格式。\n");

}

}

data\_position=false;

data\_time=false;

parking(parkingspace,car,position);

}

else if(!allparkingspaceEmpty(parkingspace)&&quEmpty(waitingqueue\_en)){

defaultInit(car);

printf("请输入车牌号码（纯数字）：");

scanf("%d",&number);

setNumber(car,number);

enQueue(waitingqueue\_en,car);

}

else{

printf("！停车场已满！请找别的停车场停车。");

}

break;

case 2:

if(quEmpty(waitingqueue\_de)){

printf("请输入要出站的车牌号码：");

scanf("%d",&number);

if(departure(parkingspace,waitingqueue\_de,number)){

if(!quEmpty0(waitingqueue\_en)){

defaultInit(car);

deQueue(waitingqueue\_en,car);

while(!data\_position||!data\_time){

printf("当前车牌号码为 %d 的汽车正要进站停车\n请输入入场停车的位置（必须为当前空车位）：",car.number);

scanf("%d",&position);

printf("请输入到达时间（请按格式输入否则后果自负）\n（！输入格式 小时：分钟 \n冒号应为中文输入法下的冒号）：");

scanf("%d：%d",&arrive\_hour,&arrive\_minute);

data\_position=setPosition(car,position);

data\_time=setArrivetime(car,arrive\_hour,arrive\_minute);

if(!data\_position||!data\_time){

printf("输入数据有误请仔细检查输入格式。\n");

}

}

data\_position=false;

data\_time=false;

parking(parkingspace,car,position);

}

}

else{

printf("！输入的车牌号码不在停车场内！");

}

}

else{

printf("出站队列已满！请先收取费用！");

}

break;

case 3:

if(!quEmpty0(waitingqueue\_de)){

defaultInit(car);

deQueue(waitingqueue\_de,car);

printf("$当前车牌号码为 %d 的汽车正要计费出站",car.number);

printf("请输入现在时间（请按格式输入否则后果自负）\n（！输入格式 小时：分钟 \n冒号应为中文输入法下的冒号）：");

scanf("%d：%d",&hour,&minute);

sum=((hour-car.arrive\_hour)\*6000+(minute-car.arrive\_minute)\*100)/(60/B);

while(hour<car.arrive\_hour||hour==car.arrive\_hour&&minute<car.arrive\_minute){

if(hour<car.arrive\_hour||hour==car.arrive\_hour&&minute<car.arrive\_minute){

printf("！请输入正确的时间！");

scanf("%d：%d",&hour,&minute);

}

sum=((hour-car.arrive\_hour)\*6000+(minute-car.arrive\_minute)\*100)/(60/B);

}

printf("$请收费：%2.4f元",sum/100);

}

else{

printf("！当前无要出站汽车！");

}

break;

case 4:

printf("谢谢使用！");

run=false;

break;

default:

printf("错误！：请输入正确的功能序号。\n");

break;

}

}

return 0;

}