#include<iostream>

#include<stdio.h>

#include<stack>

#include<list>

#include<map>

using namespace std;

typedef struct NODE

{

char chepai[7];

int id;

int hourB;

int minuteB;

int hourL;

int minuteL;

}CAR;

list <CAR> carLev;

list <CAR> carW;

list <CAR> carD;

stack <CAR> carT;

int m\_shuMn = 0;

int m\_shu = 1;

int m\_id = 0;

char GetKey(); //获取键盘信息

int GetNum(); //获取车库大小

void SaveCar(); //车辆进入时间及信息

void InputCarT(CAR &input); //停车场车辆

void InputCarD(CAR &input); //便道车辆

void SetLevTime(); //车辆离开时间

void SetNewCar();

void GetLevCar(); //离开车辆信息

void ChangeCarT();

void ShowALLCar(); //停车场以及便道车辆信息

int main()

{

char c;

while(1)

{

printf(" ——————————————————\n");

printf("| 1.停车场可停车辆数目 |\n");

printf("| 2.车辆进入时间及信息 |\n");

printf("| 3.车辆离开时间 |\n");

printf("| 4.离开车辆信息 |\n");

printf("| 5.停车场以及便道车辆信息 |\n");

printf("| q.退出 |\n");

printf(" ——————————————————\n");

c = GetKey();

switch(c)

{

case '1':

m\_shuMn = GetNum();

break;

case '2':

SaveCar();

break;

case '3':

SetLevTime();

SetNewCar();

break;

case '4':

GetLevCar();

break;

case '5':

ShowALLCar();

break;

case 'q':

return 0;

break;

}

}

return 0;

}

char GetKey()

{

char c;

char z;

int flag = 1;

while((c = getchar()) != '\n' || 1 == flag)

{

z = c;

flag = 0;

}

return z;

}

int GetNum()

{

int n;

printf("请输入停车车场可停车辆数目\n");

scanf("%d",&n);

return n;

}

void SaveCar()

{

CAR input;

if(m\_shu <= m\_shuMn)

InputCarT(input);

else

{

printf("停车场已经停满\n");

InputCarD(input);

}

}

void InputCarT(CAR &input)

{

printf("请输入车牌号 :\n");

for(int i = 0;i <= 7;i ++)

{

scanf("%c",&input.chepai[i]);

}

printf("\n");

printf("请输入时间 时:");

scanf("%d",&input.hourB);

printf(" 分:");

scanf("%d",&input.minuteB);

printf("\n");

input.id = m\_shu;

carT.push(input);

printf("您的车位在北·第%d号位置",m\_shu);

m\_shu++;

}

void InputCarD(CAR &input)

{

printf("请输入车牌号 :");

for(int i = 0;i <= 7;i ++)

{

scanf("%c",&input.chepai[i]);

}

printf("\n");

printf("请输入时间 时:");

scanf("%d",&input.hourB);

printf(" 分:");

scanf("%d",&input.minuteB);

printf("\n");

input.id = m\_shu;

carD.push\_back(input);

m\_shu++;

}

void SetLevTime()

{

printf("离开车辆停靠位置 :");

int id;

scanf("%d",&id);

while(!carT.empty())

{

if(carT.top().id == id)

{

CAR input;

printf("离开请输入时间 时:");

scanf("%d",&input.hourL);

printf(" 分:");

scanf("%d",&input.minuteL);

printf("\n");

for(int i = 0;i <=7;i ++)

{

input.chepai[i] = carT.top().chepai[i];

}

m\_id = carT.top().id;

input.hourB = carT.top().hourB;

input.minuteB = carT.top().minuteB;

carLev.push\_back(input);

m\_shu--;

carT.pop();

break;

}

else

{

CAR input;

for(int i = 0;i <=7;i ++)

{

input.chepai[i] = carT.top().chepai[i];

}

input.id = carT.top().id;

input.hourB = carT.top().hourB;

input.minuteB = carT.top().minuteB;

carW.push\_back(input);

}

carT.pop();

}

}

void SetNewCar()

{

//reverse(carW.begin(),carW.end());

if(carW.empty() != true)

{

list<CAR> ::iterator ite = carW.begin();

while(ite != carW.end())

{

CAR input;

for(int i = 0;i <=7;i ++)

{

input.chepai[i] = ite->chepai[i];

}

input.id = ite->id;

input.hourB = ite->hourB;

input.minuteB = ite->minuteB;

carT.push(input);

ite++;

}

//list<CAR> ::iterator ite1 = carW.begin();

//while(ite1 != carW.end())

//{

// //删除对象

// delete (ite1);

// //删除节点，返回下一个

// ite1 = carW.erase(ite1);

//}

carW.clear();

}

if(carD.empty() != true) //判断链表是否为空

{

CAR input;

for(int i = 0;i <=7;i ++)

{

input.chepai[i] = carD.begin()->chepai[i];

}

input.id = m\_id;

input.hourB = carD.begin()->hourB;

input.minuteB = carD.begin()->minuteB;

carT.push(input);

carD.pop\_front();

}

}

void GetLevCar()

{

int n,n1;

// 遍历链表

list<CAR> ::iterator ite = carLev.begin();

while(ite != carLev.end())

{

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

printf("车辆车牌号 :");

for(int i = 0;i <=7;i ++)

{

printf("%c",ite->chepai[i]);

}

printf("\n");

printf("车辆进入时间 时：%d 分：%d\n",ite->hourB,ite->minuteB);

printf("离开时间 时：%d 分：%d\n",ite->hourL,ite->minuteL);

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

if(ite->minuteB < ite->minuteL)

{

n = ite->minuteL - ite->minuteB;

}

else

n = ite->minuteB - ite->minuteL;

n = n + ite->hourL - ite->hourB;

printf("停留时间 （分钟）：%d\n",n);

n1 = n\*2;

printf("应缴费用 (2元 /每分钟)：%d\n",n1);

++ite;

}

}

void ShowALLCar()

{

ChangeCarT();

int id;

printf("-------------停车场内车辆--------------\n");

if(carT.empty() != true) //判断链表是否为空

{

while(!carT.empty())

{

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

printf("车辆车牌号 :");

for(int i = 0;i <=7;i ++)

{

printf("%c",carT.top().chepai[i]);

}

printf("\n");

printf("车辆进入时间 时：%d 分：%d\n",carT.top().hourB,carT.top().minuteB);

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

CAR input;

for(int i = 0;i <=7;i ++)

{

input.chepai[i] = carT.top().chepai[i];

}

input.id = carT.top().id;

input.hourB = carT.top().hourB;

input.minuteB = carT.top().minuteB;

carW.push\_back(input);

carT.pop();

}

list<CAR> ::iterator ite1 = carW.begin();

while(ite1 != carW.end())

{

CAR input;

for(int i = 0;i <=7;i ++)

{

input.chepai[i] = ite1->chepai[i];

}

input.id = ite1->id;

input.hourB = ite1->hourB;

input.minuteB = ite1->minuteB;

carT.push(input);

ite1++;

}

carW.clear();

ChangeCarT();

}

// 遍历链表

printf("-------------便道等待车辆--------------\n");

list<CAR> ::iterator ite = carD.begin();

while(ite != carD.end())

{

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

printf("车辆车牌号 :");

for(int i = 0;i <=7;i ++)

{

printf("%c",ite->chepai[i]);

}

printf("\n");

printf("车辆进入时间 时：%d 分：%d\n",ite->hourB,ite->minuteB);

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

++ite;

}

ChangeCarT();

}

void ChangeCarT()

{

while(!carT.empty())

{

CAR input;

for(int i = 0;i <=7;i ++)

{

input.chepai[i] = carT.top().chepai[i];

}

input.id = carT.top().id;

input.hourB = carT.top().hourB;

input.minuteB = carT.top().minuteB;

carW.push\_back(input);

carT.pop();

}

list<CAR> ::iterator ite1 = carW.begin();

while(ite1 != carW.end())

{

CAR input;

for(int i = 0;i <=7;i ++)

{

input.chepai[i] = ite1->chepai[i];

}

input.id = ite1->id;

input.hourB = ite1->hourB;

input.minuteB = ite1->minuteB;

carT.push(input);

ite1++;

}

carW.clear();

}