**《仿真电梯》实习报告**

题目：设计一个电梯模拟系统

班级：19184115

专业：计算机科学与技术（人工智能与大数据）创新实验班

姓名：洪智豪 学号：19195216

完成日期：2020.11.11

1. **需求分析**
2. 电梯控制
3. 电梯初始在1层
4. 当没有楼层有用户请求时，经过300秒后，电梯将回到1层待命
5. 电梯在超载或者长时间停留在除1层外的某层后，将会报警。
6. 电梯在超重后，将会报警
7. 电梯在上升状态时，若楼上还有人有请求或者电梯里还有人，继续保持上升状态；若没有楼层有请求并且电梯里没人，切换为等待状态；若楼下还有人有请求并且电梯里没人，切换为下降状态。
8. 电梯在下降状态时，若楼下还有人有请求或者电梯里还有人，继续保持下降状态；若没有楼层有请求并且电梯里没人，切换为等待状态；若楼上还有人有请求并且电梯里没人，切换为上升状态。
9. 每个人有一个能容忍的最大等待时间，一旦电梯等候时间过长，会放弃
10. 当电梯超载时，部分乘客将无法进入
11. 乘客将按照先到先进的顺序进入电梯，如果某个乘客因为电梯超重无法进入，则后面的人仍可进入电梯。
12. 可以通过文件导入测试数据
13. 时钟从0开始，时间单位为1秒，人和电梯各种动作均要耗费一定时间

例如：

电梯关门开门各用5秒

电梯上升下降一层各用5秒

每个人进出电梯需要10秒

如果电梯在某层静止时间超过300秒，则驶回1层待命

1. 电梯每经过100秒左右，用户可以键入任意键手动添加乘客信息
2. 动态显示系统状态的变化过程
3. 动态显示人的进出

每隔一秒钟刷新一次界面，人前进或后退一格。

1. 动态显示电梯的升降

每隔一秒种刷新一次界面，电梯上升或下降一格

1. 动态显示电梯的报警

报警或取消报警时，刷新一次界面

1. **概要设计**
2. 数据结构设计
3. 链队列LinkQueue

用于装载用户对象的链队列

1. 结构体定义

typedef struct QNode

{

Person data;

structQNode\* next;

}QNode, \* QueuePtr;

typedef struct LinkQueue

{

QueuePtr front;

QueuePtr rear;

}LinkQueue;

1. 初始化队列

Status InitQueue(LinkQueue &Q)

Q为待初始化的队列参数

若成功返回OK

失败返回OVERFLOW

1. 判断队列是否为空

Status IsQueueEmpty(LinkQueue &Q)

Q为待初始化的队列参数

若为空返回OK

若非空返回 0

1. 销毁队列

Status DestroyQueue(LinkQueue &Q)

Q为待初始化的队列参数

若成功返回OK

1. 元素入队

Status EnQueue(LinkQueue &Q, Persone)

Q为待初始化的队列参数e为待入队的元素

若成功返回OK

1. 元素出队

Status DeQueue(LinkQueue &Q, Persone)

Q为待初始化的队列参数 e为出队的元素

若成功返回OK

否则返回ERROR

1. 电梯Elevator
2. int NowFloor电梯现在所在的楼层
3. int MaxFloor最高能到几层
4. int NowWeight电梯现在的载重
5. int MaxWeight电梯的最大载重
6. intNowWaitTime;//当前楼层等待时间
7. int MaxWaitTime
8. int NowStayTime; //无人使用的等待时间
9. int MaxStayTime;
10. int NowPassengerNumber;
11. int MaxPassengerNumber
12. int Mode; //0等待，1上升，2下降
13. int WeightWarn;//1为报警，0为正常状态
14. int TimeWarn;//1为报警，0为正常状态
15. LinkQueueUpQueue电梯中需要上楼的乘客所在的上升队列
16. LinkQueueDownQueue电梯中需要下楼的乘客所在的下降队列
17. 乘客 Passenger
18. Int InitialFloor; 乘客的起始楼层
19. Int TargetFloor乘客的目标楼层
20. Int Weight 乘客的重量
21. Int Patient 乘客的耐心值若超出 则该乘客会改走楼梯
22. IntInitialTime乘客进/出电梯所需的时间
23. 楼层Floor

IntUpNum//该楼层上升队列的人数

Int DownNum//该楼层下降队列的人数

LinkQueueUpQueue该楼层待上楼乘客排队的上升队列

LinkQueueDownQueue该楼层待下楼乘客排队的下降队列

1. 模块设计

全局变量

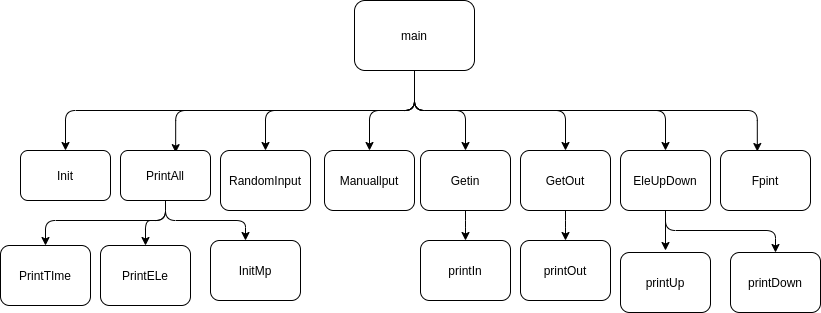
int AboveRequest; //(当前层，最高层]的请求人数

int UnderRequest; //[最底层，当前层)的请求人数

int Time; //总时间

Floor floor[15];//楼层

1. 模块结构图



1. 函数说明

项目中包含了以下函数：

1. void Init(Elevator \*ele)

ele:电梯

初始化电梯以及楼层

1. void PrintAll(Elevator \*ele)

ele:电梯

将电梯和楼层中的人以及电梯的运行状况全部打印

1. void PrintTime()

打印截止目前系统已经运行的时间

1. void PrintEle(Elevator \*ele)

ele:电梯

打印电梯 楼层 以及所有乘客的情况

1. void RandomInput(Elevator \*ele)

ele：电梯

在电梯运行的同时随机生成乘客

1. void InitMp(Elevator \*ele)

ele:电梯

初始化控制打印电梯的mp

1. void ManualInput(Elevator \*ele)

ele:电梯

手动输入乘客信息后再次启动电梯

1. void GetIn(Elevator \*ele, Floor \*flo)

ele:电梯 flo:楼层

让楼层里的人进入电梯

1. void printIn(Elevator \*ele, Passenger p)

ele:电梯 Passenger:乘客

动态显示乘客进入电梯的情况

1. void printIn(Elevator \*ele, Passenger p)

ele:电梯 Passenger:乘客

动态显示乘客进入电梯的情况

1. void GetOut(Elevator \*ele, Floor \*flo)

ele:电梯 flo:楼层

电梯中的人离开电梯进入楼层

1. void printOut(Elevator \*ele,Passenger p)

ele:电梯p:乘客

动态显示乘客从电梯出到楼层的情况

1. void EleUpDown(Elevator \*ele)

ele:电梯

控制电梯切换上升/下降/等待模式

1. void printUp(Elevator \*ele,Passenger p)

ele:电梯 p:乘客

动态显示电梯上升

1. void printDown(Elevator \*ele,Passenger p)

ele:电梯 p:乘客

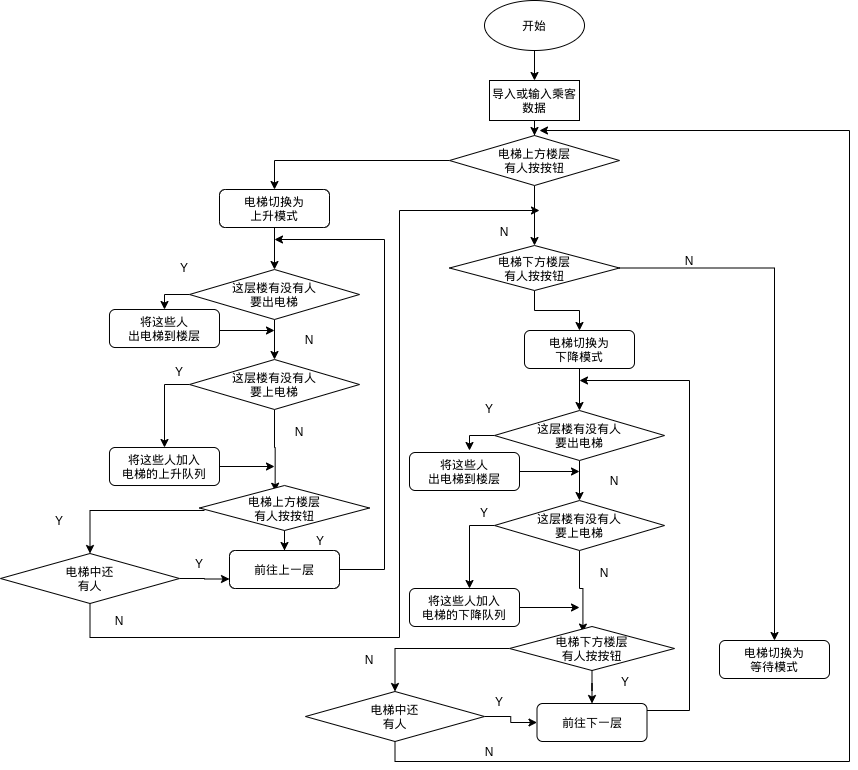
动态显示电梯下降

1. void Fprint(Elevator \*ele)

ele:电梯

通过文件导入的方式向楼层中写入乘客数据

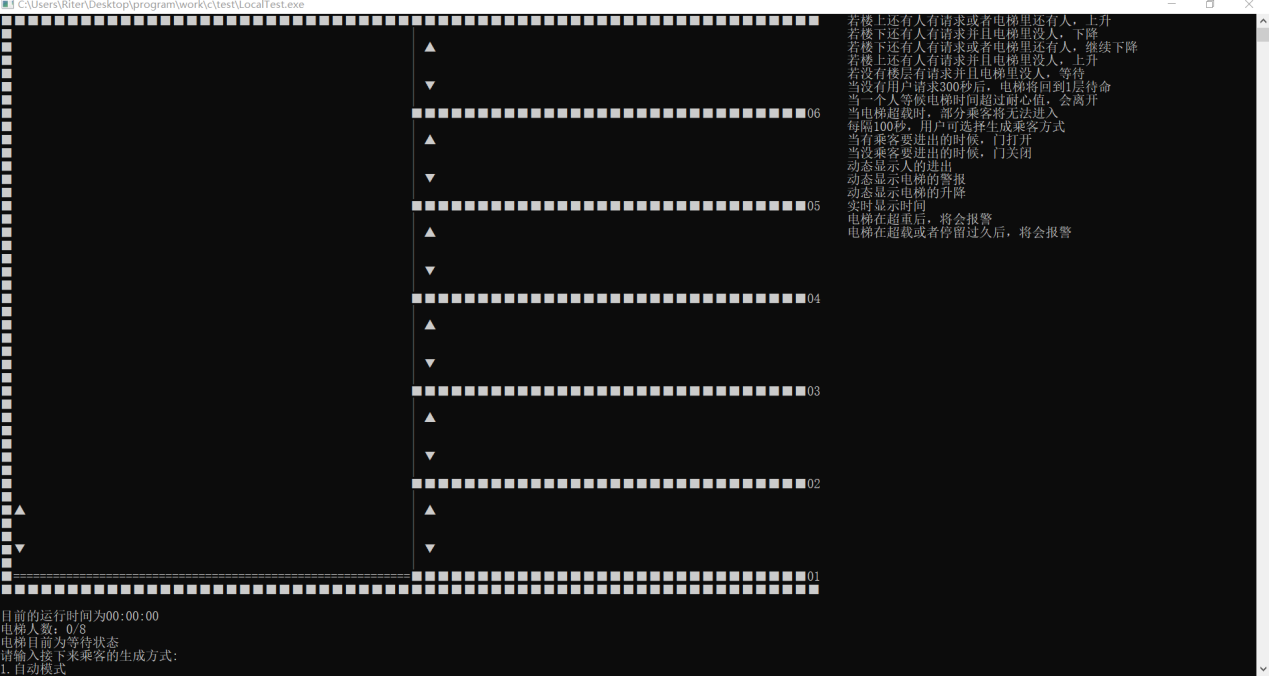
1. **详细设计**
2. Main函数程序流程图



1. **用户手册**

用户界面

1. 输入1自动生成乘客并启动电梯
2. 输入2由用户手动输入乘客信息 并启动电梯
3. 100秒后电梯会暂停等待输入
4. 规则显示在楼层右侧，句子左侧打勾为目前使用规则，右侧为是否使用过



**五** ．**测试分析**

* 1. 通过自动生成或者手动输入生成乘客

1. 通过手动输入来生成乘客并启动电梯

输入用例：

2

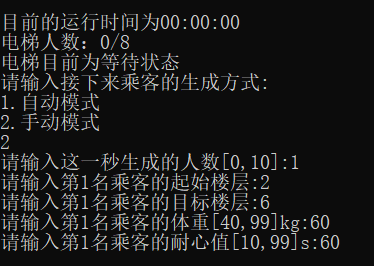
1

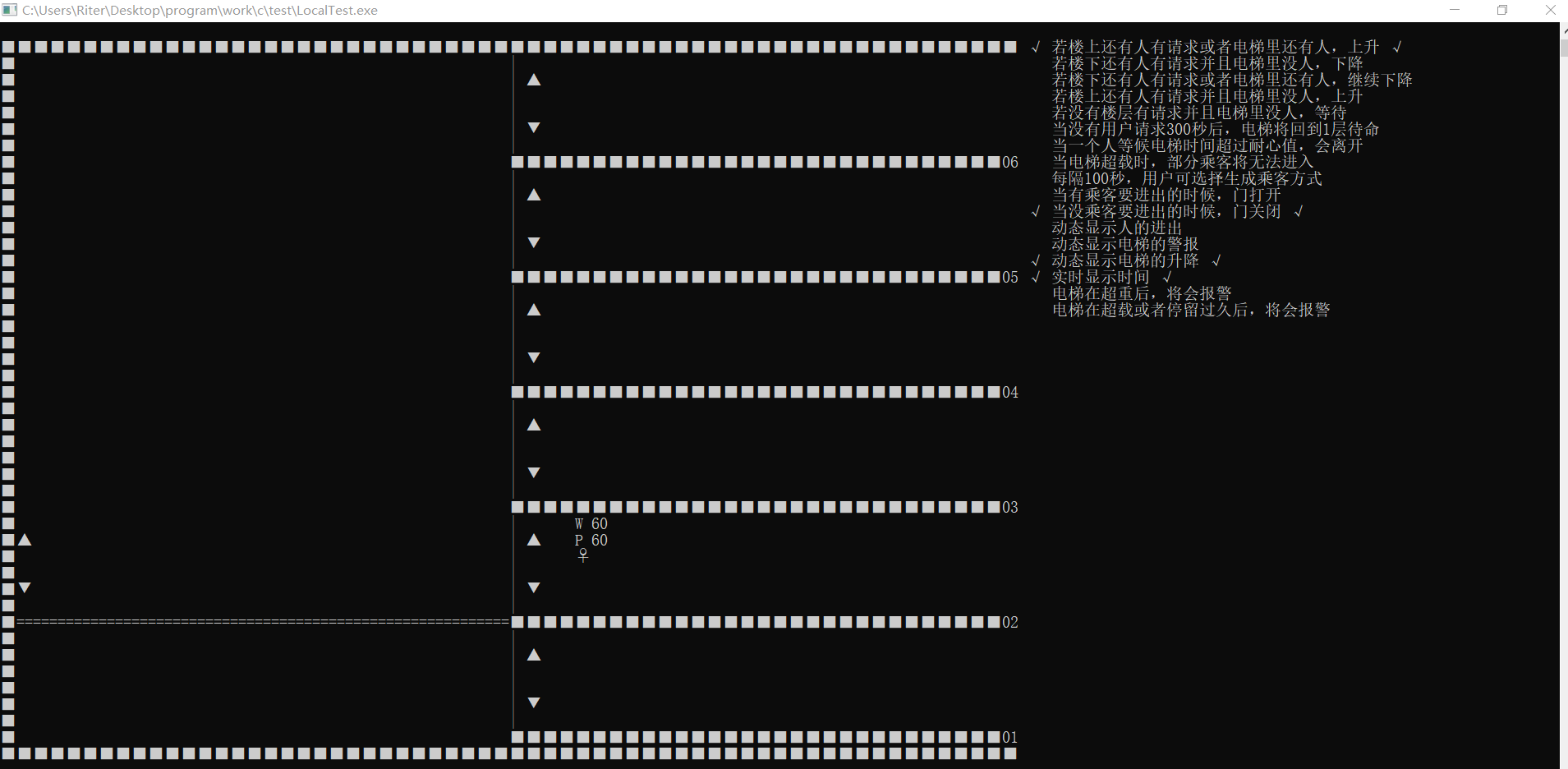
2 6 60 60

期望结果：

在2楼生成一个目标为6楼体重和耐心值均为60的乘客 并启动电梯

实测结果：





结论：测试通过

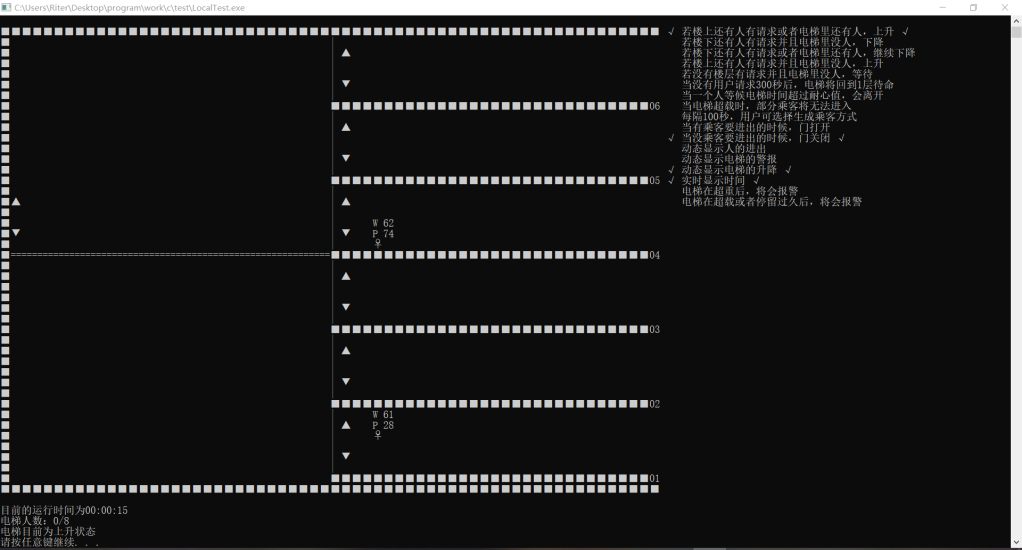
1. 通过自动生成来生成乘客并启动电梯
2. 输入用例：

1

期望结果：

电梯在启动的同时生成乘客

1. 实测结果



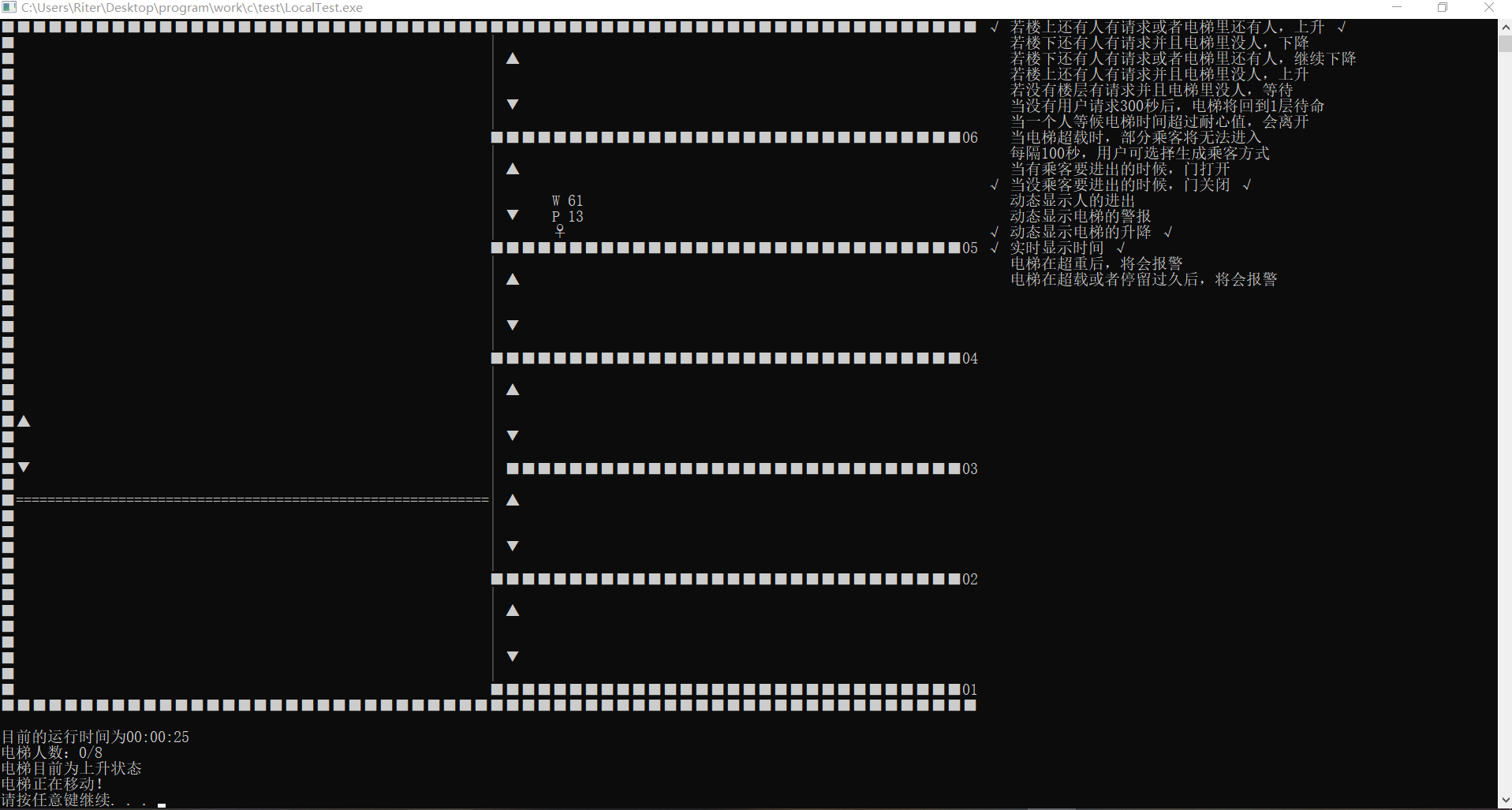
1. 结论：测试通过
   1. 仿真电梯

以下测试均通过随机生成乘客测试（顺便测试程序的稳定性）

期望结果 右侧菜单区相应位置打勾

1. 电梯在上升状态时 若楼上还有人有请求或者电梯里还有人，继续上升

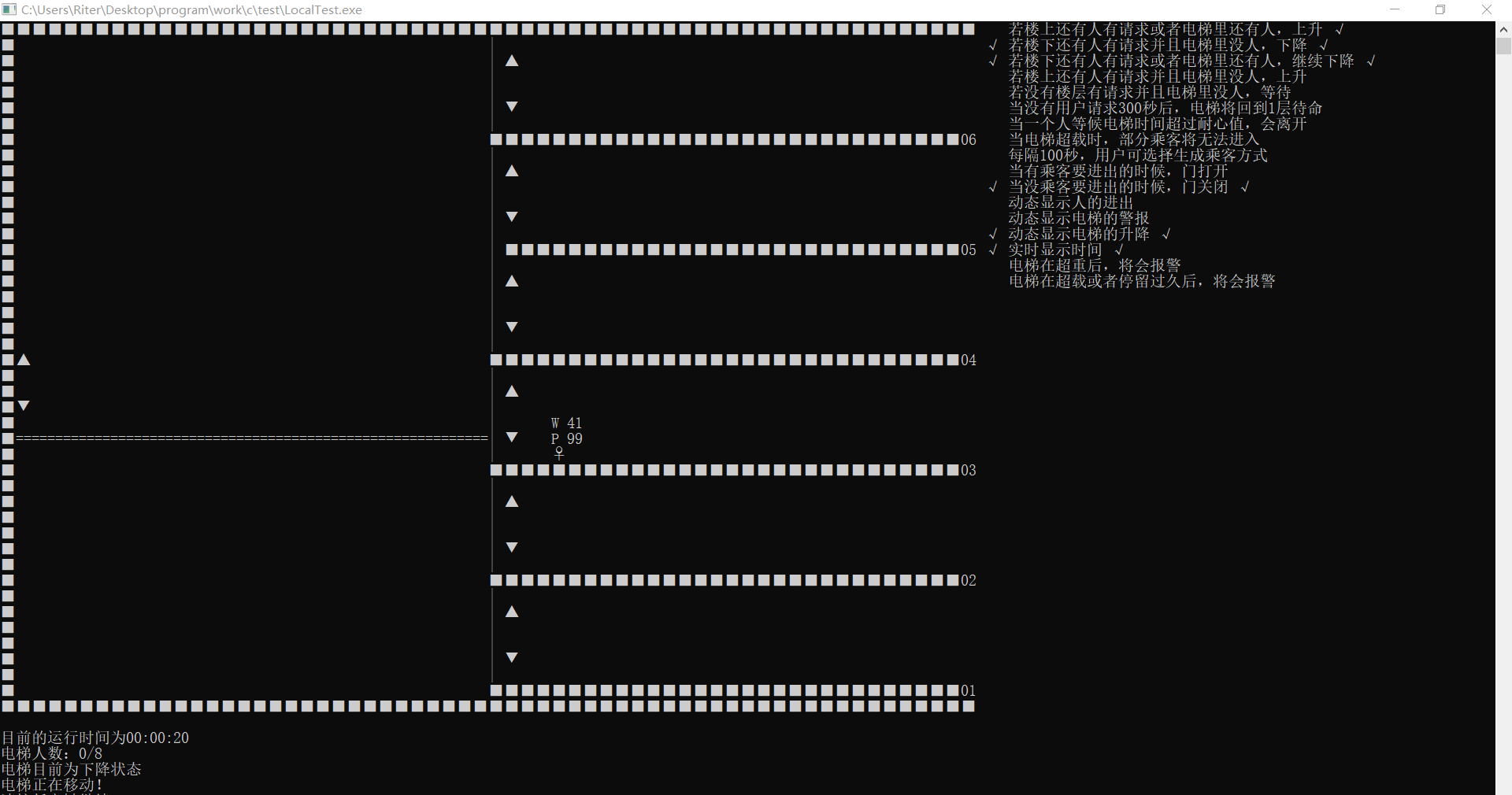
2)实测结果



3)结论：测试通过

1. 若楼下还有人有请求并且电梯里没人，切换为下降状态

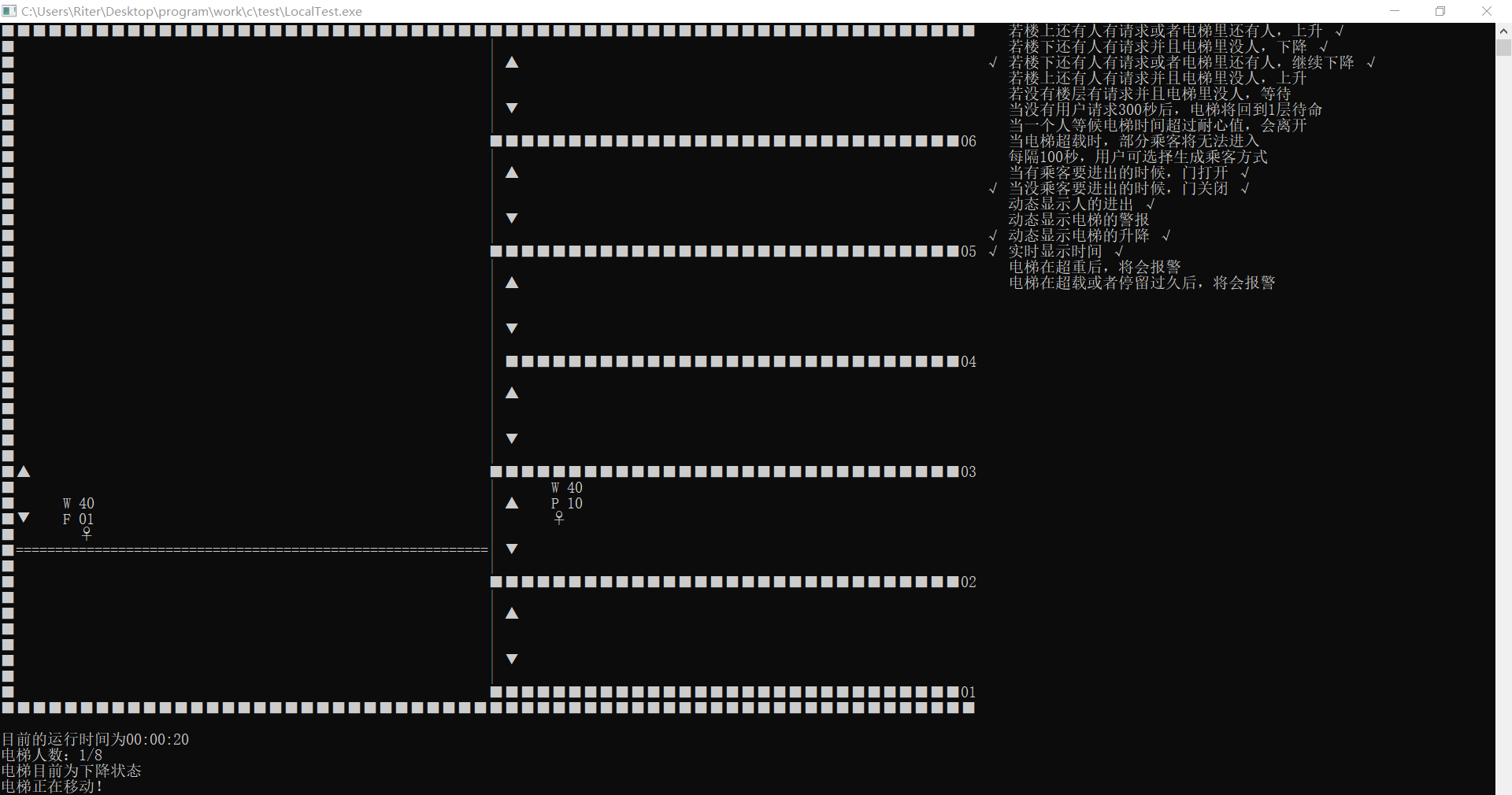
2)实测结果



3)结论：测试通过

1. 电梯在下降状态时 若楼下还有人有请求或者电梯里还有人，继续下降

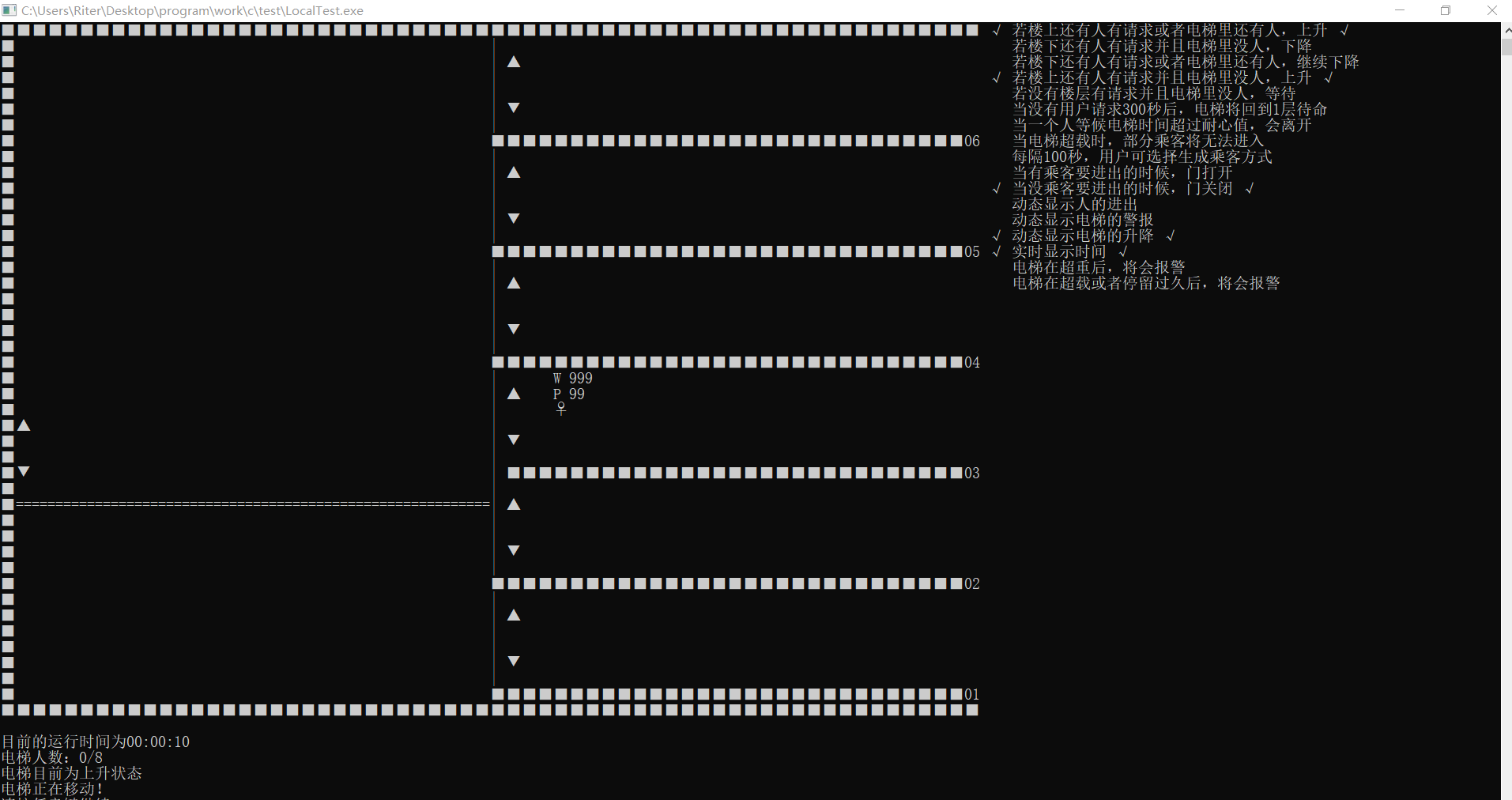
2)实测结果



3)结论：测试通过

1. 若楼上还有人有请求并且电梯里没人，切换为上升状态

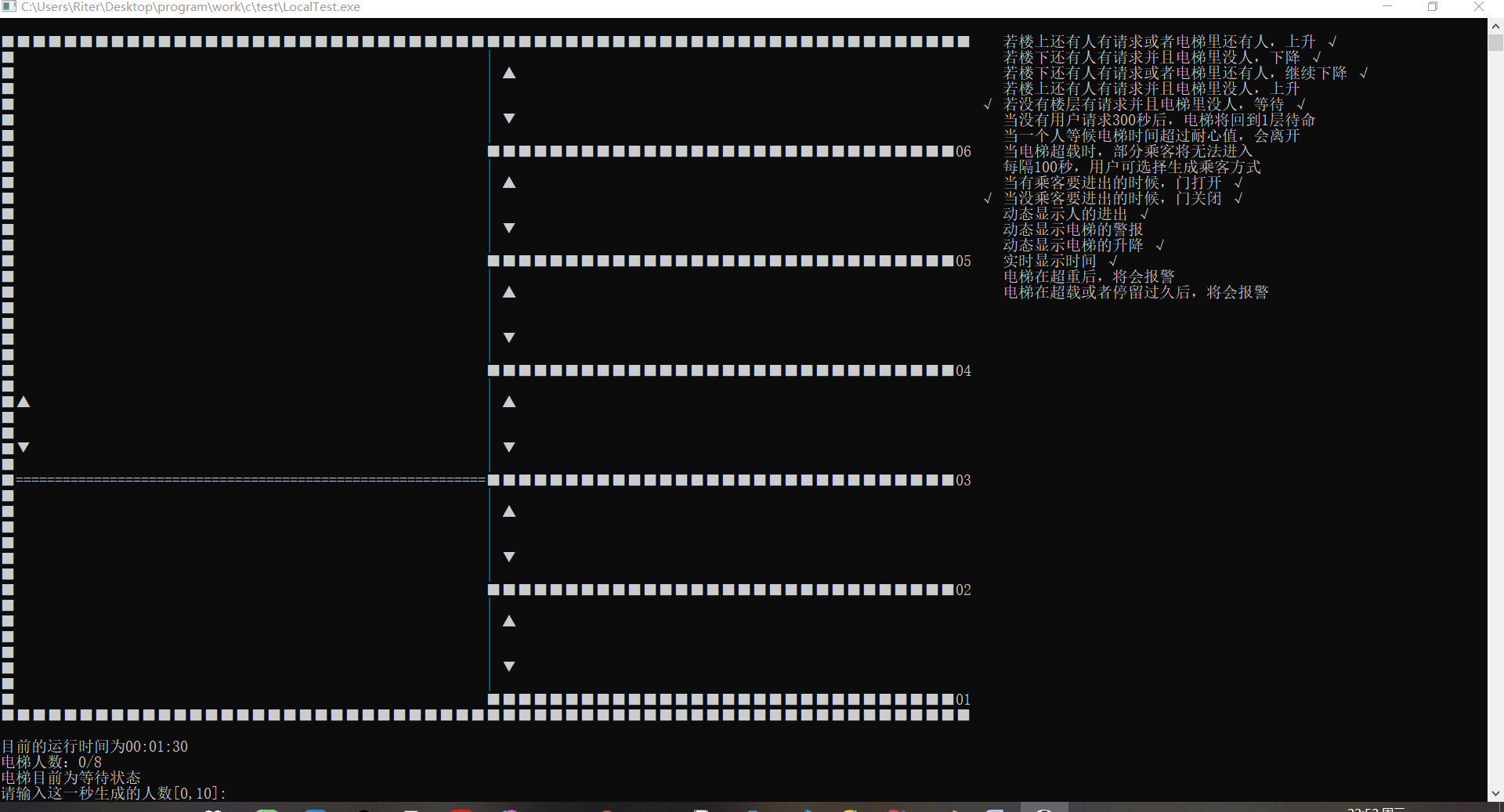
2)实测结果



3)结论：测试通过

1. 若没有楼层有请求并且电梯里没人，切换为等待状态

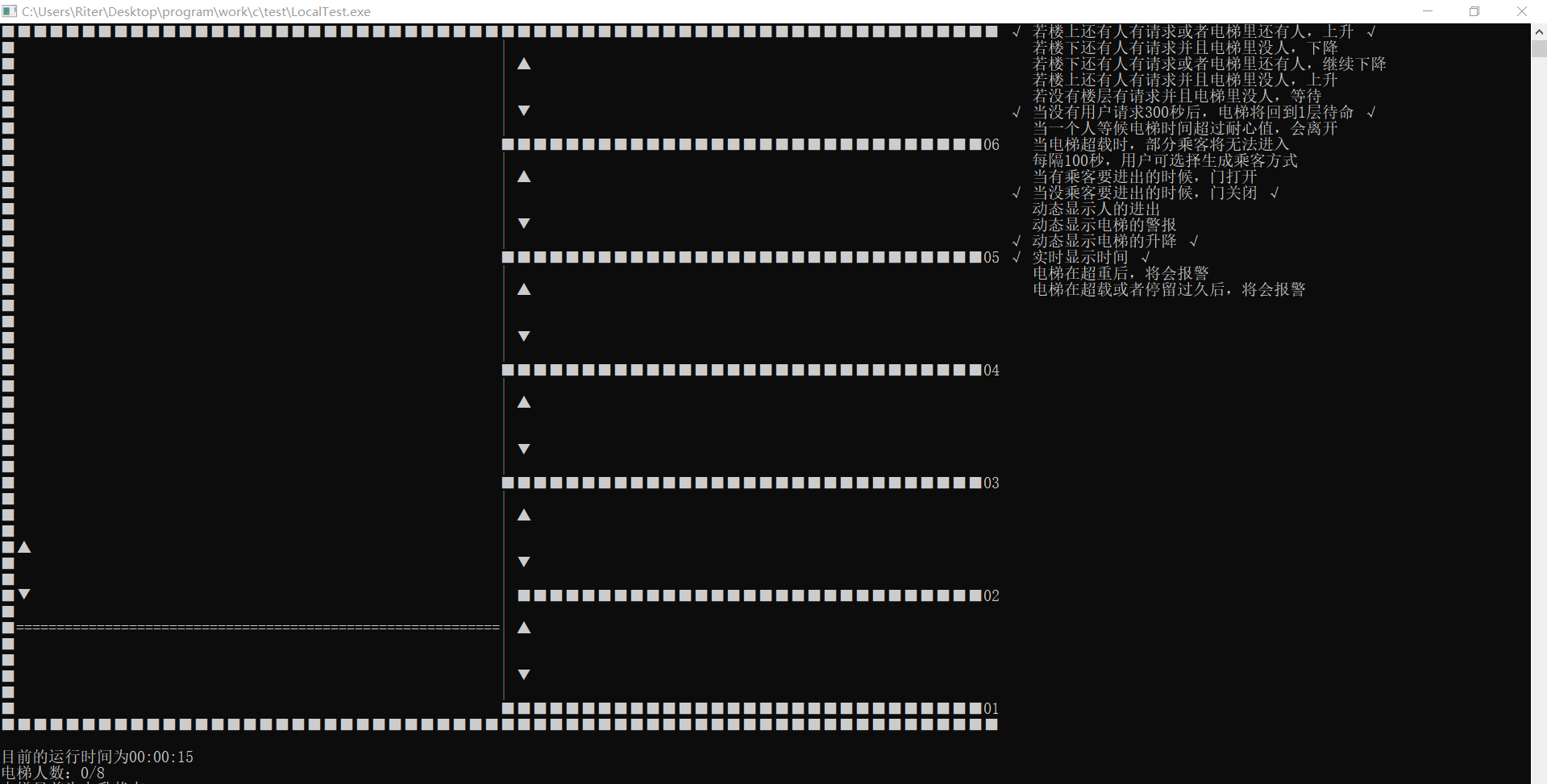
2)实测结果



3)结论：测试通过

1. 当没有用户请求时，300秒后，电梯将回到1层待命

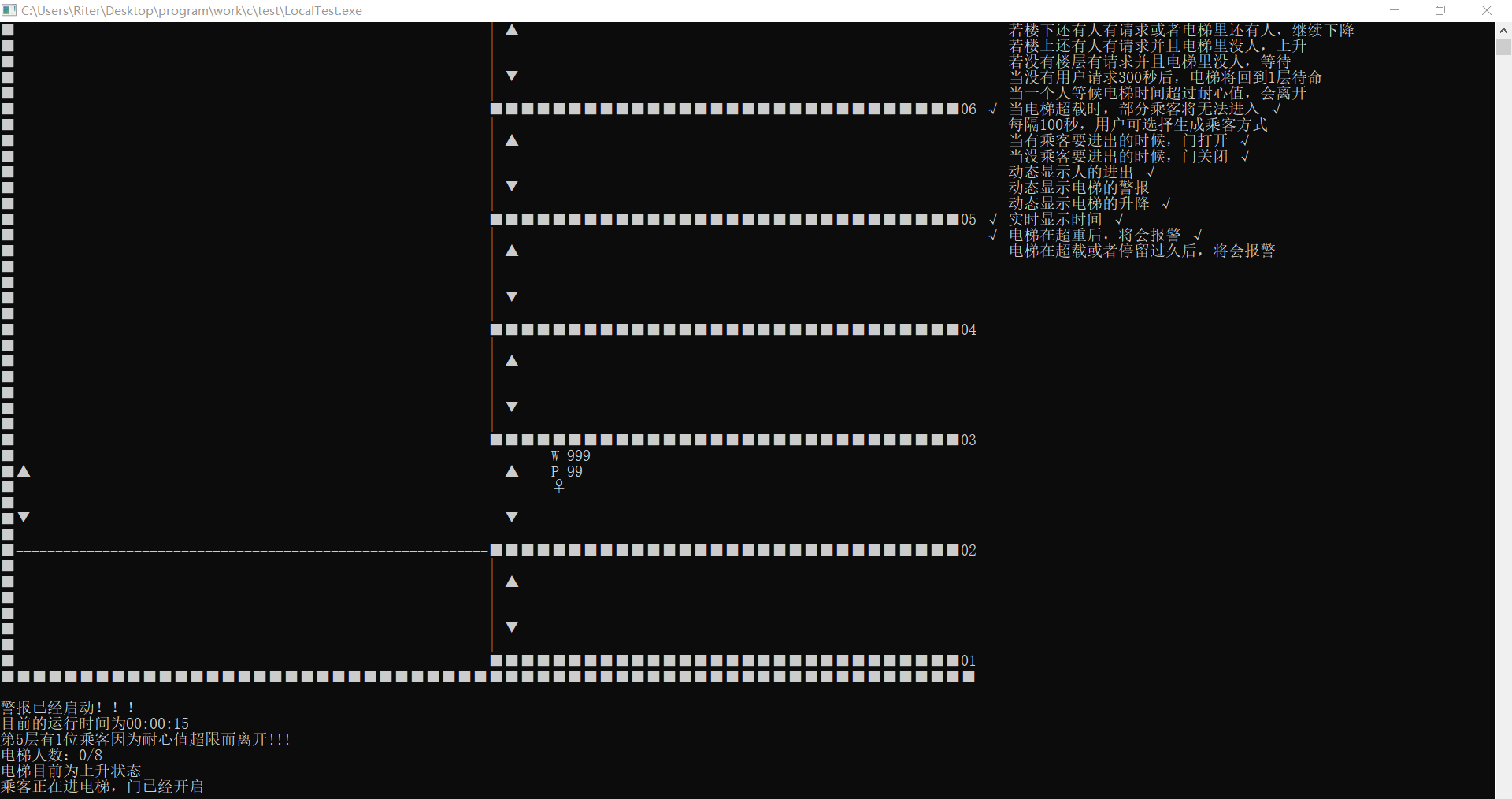
2)实测结果



3)结论：测试通过

1. 当一个人等候电梯时间超过耐心值，会离开

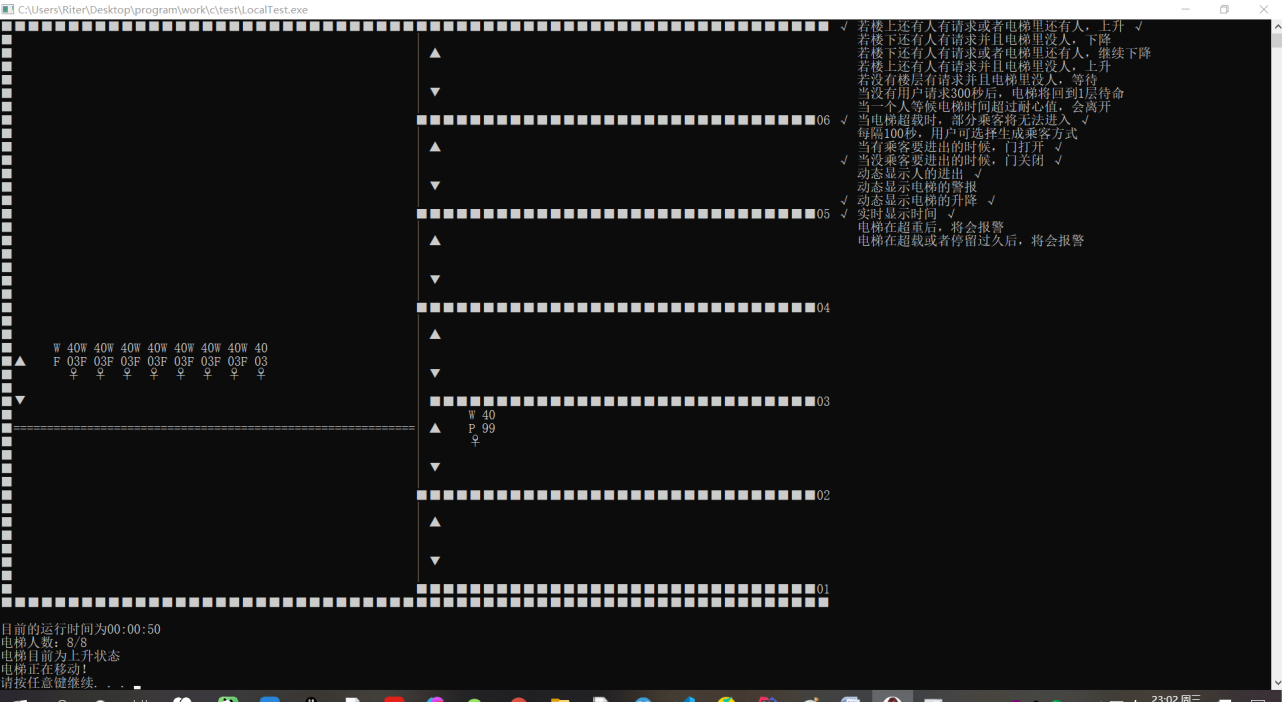
2)实测结果



3)结论：测试通过

1. 当电梯超载时，部分乘客将无法进入

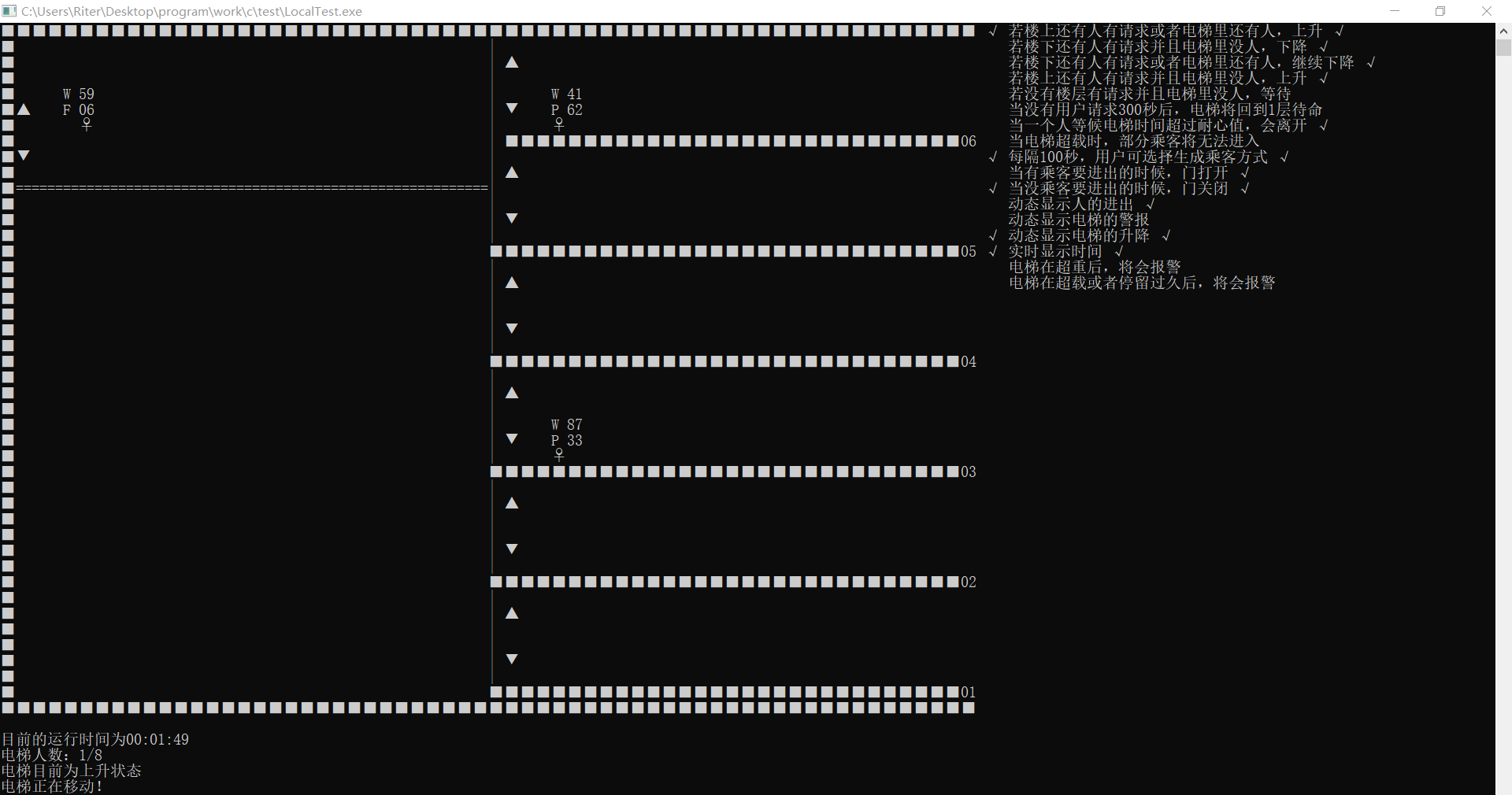
2)实测结果



3)结论：测试通过

1. 每隔100秒，用户可选择自动生成或者手动添加乘客

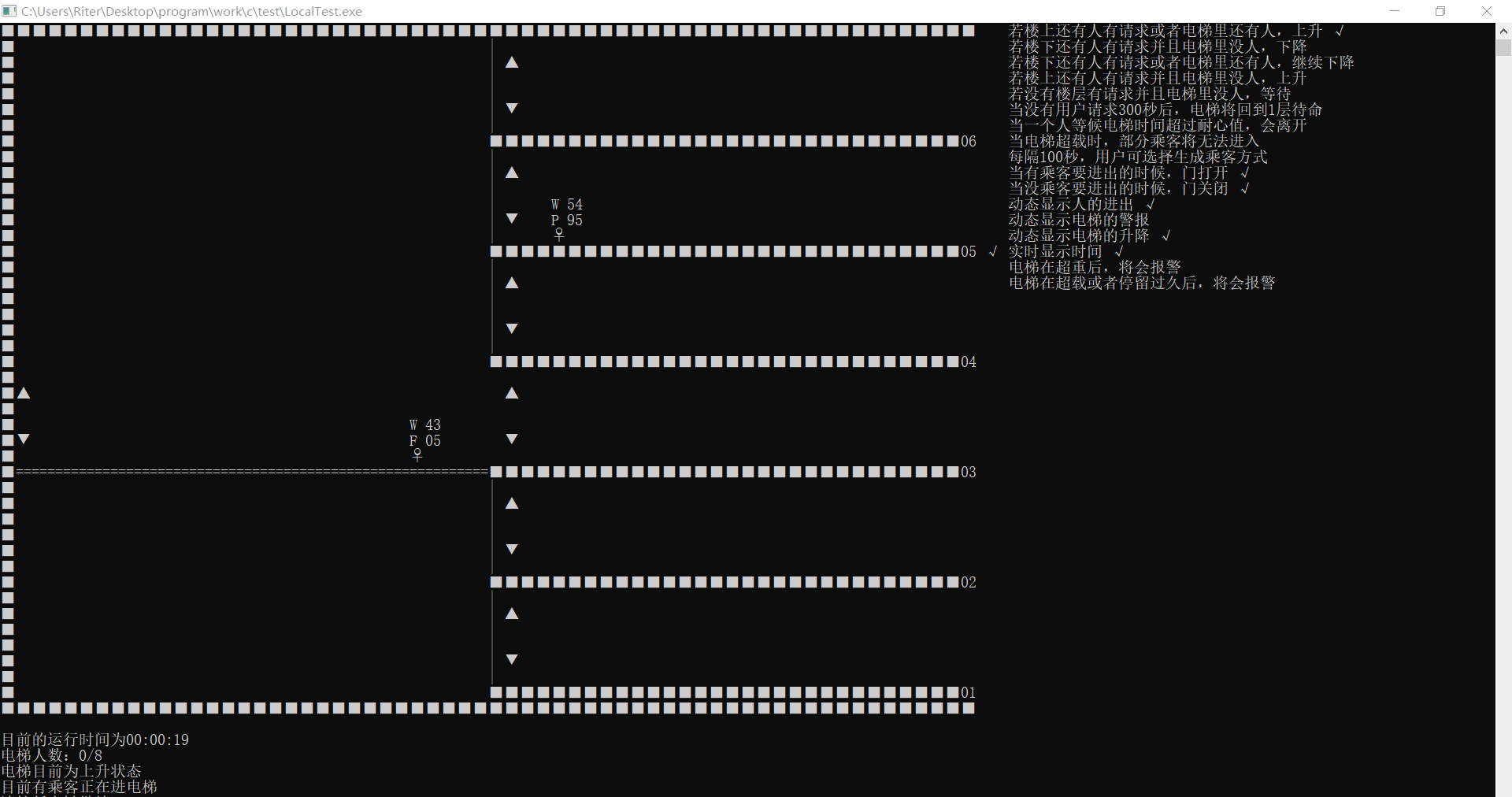
2)实测结果



3)结论：测试通过

1. 当有乘客要进出的时候，门打开

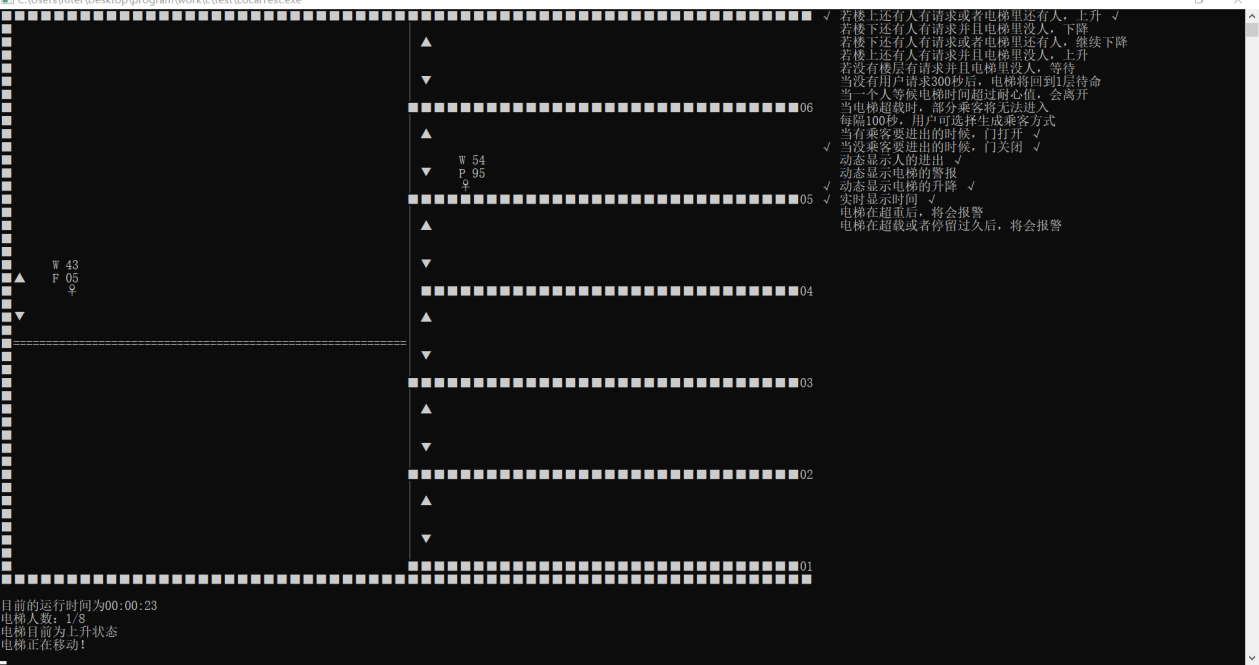
2)实测结果



3)结论：测试通过

1. 当没乘客要进出的时候，门关闭

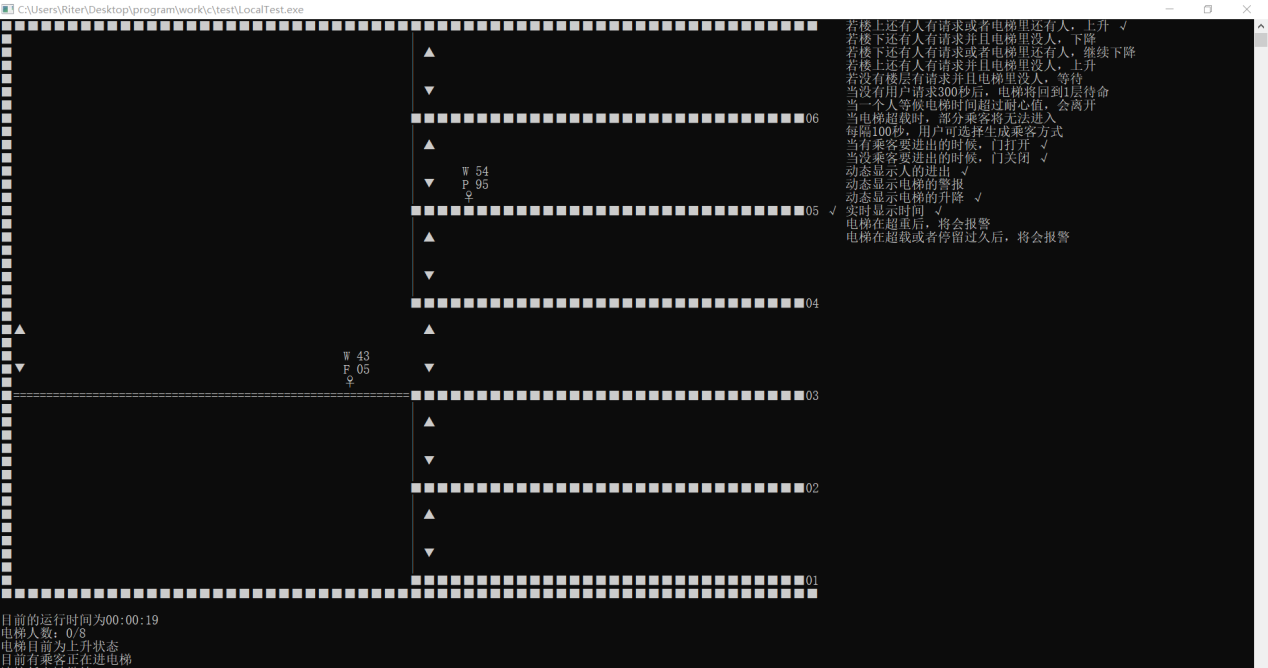
2)实测结果



3)结论：测试通过

1. 动态显示人的进出

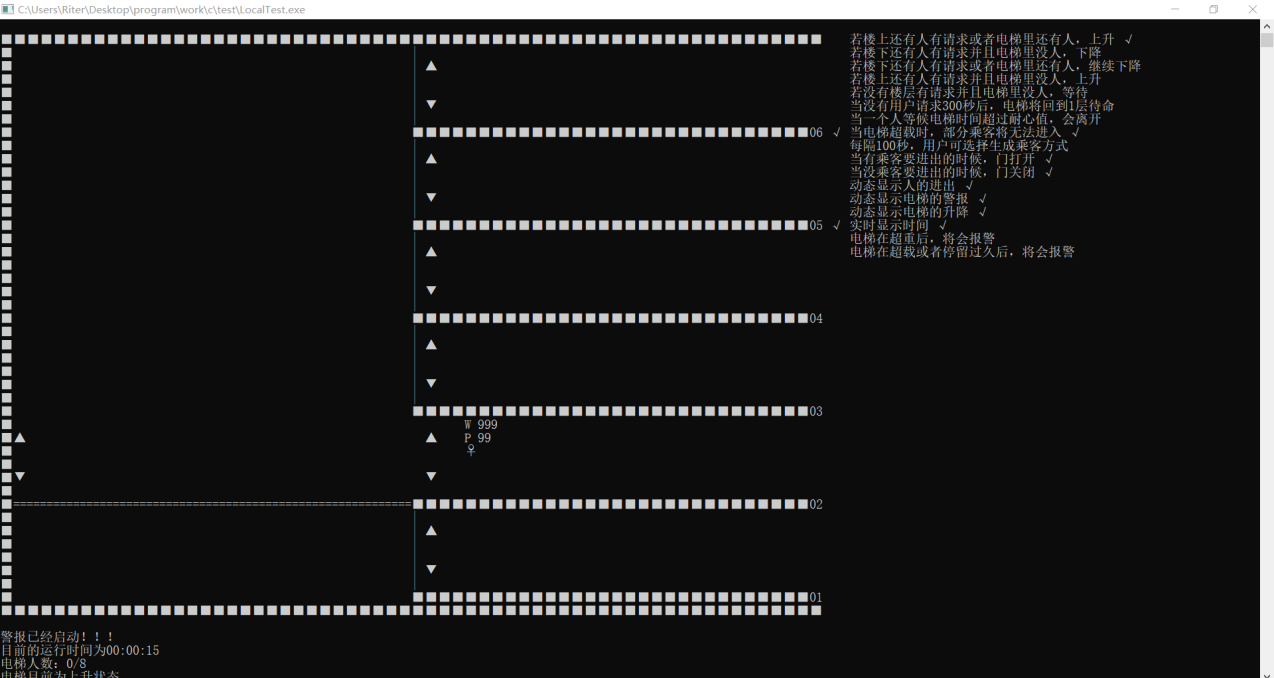
2)实测结果



3)结论：测试通过

1. 动态显示电梯的警报

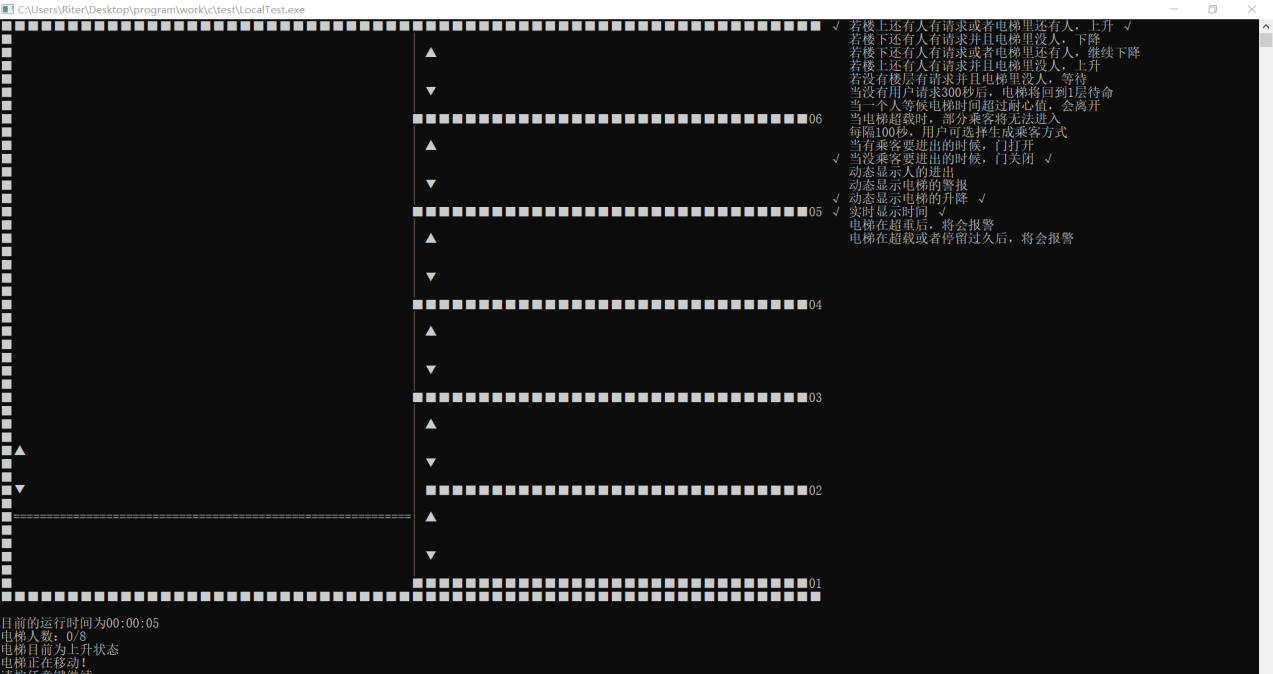
2)实测结果



3)结论：测试通过

1. 动态显示电梯的升降

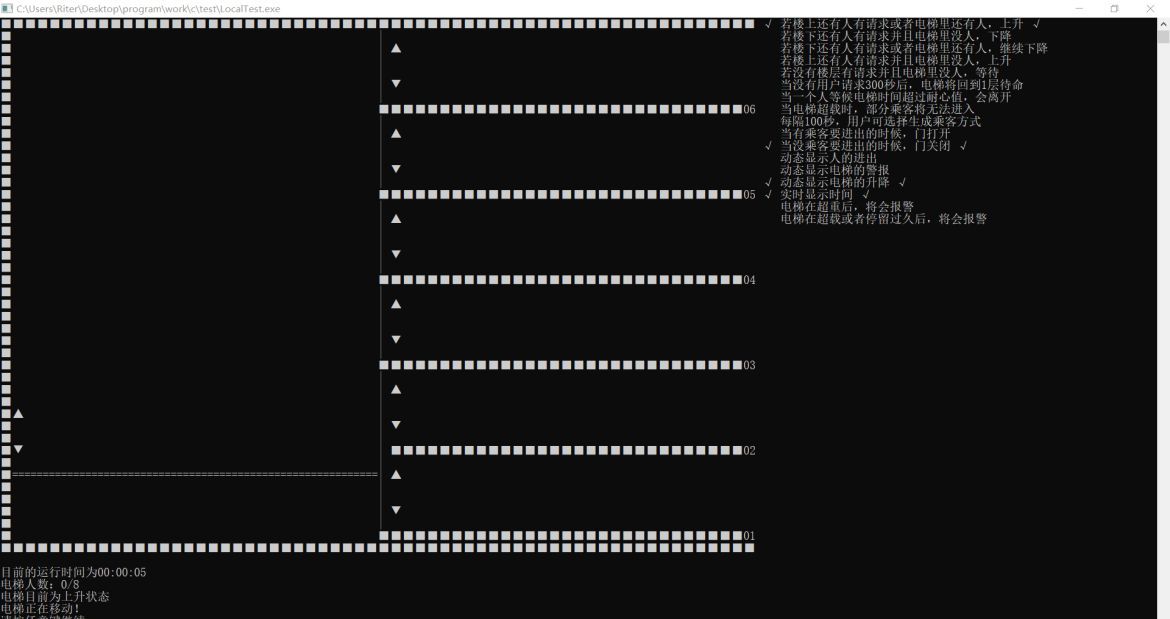
2)实测结果



3)结论：测试通过

1. 实时显示时间

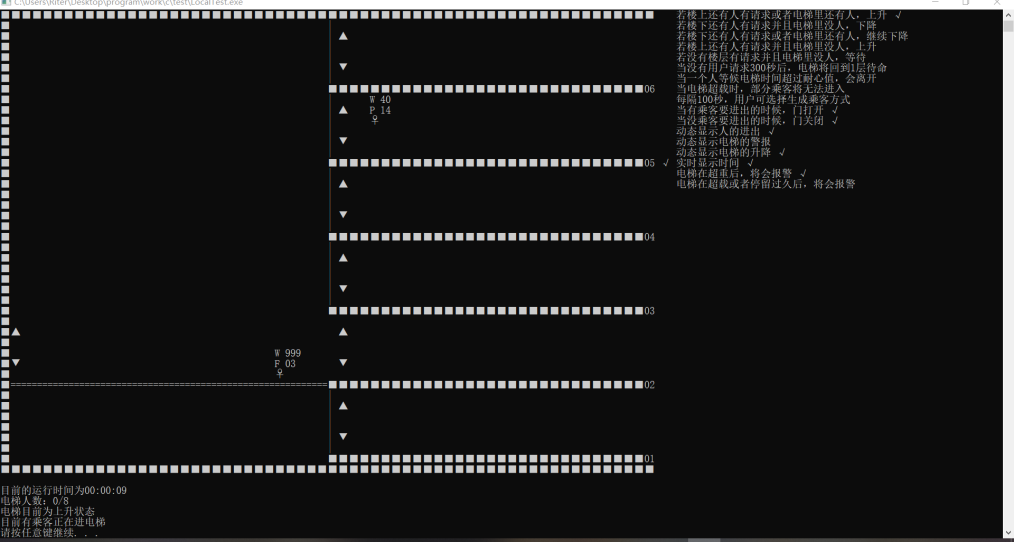
2)实测结果

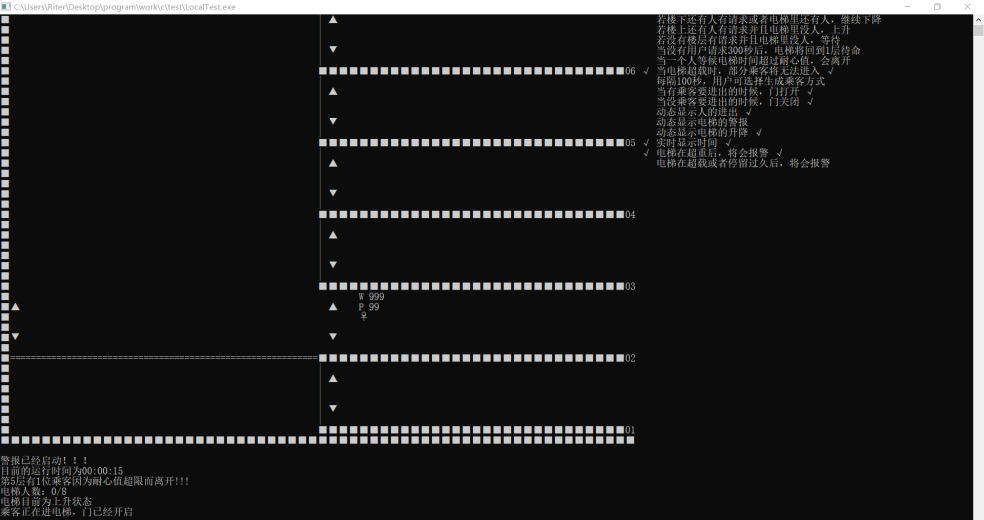


3)结论：测试通过

1. 电梯在超重后，将会报警

2)实测结果

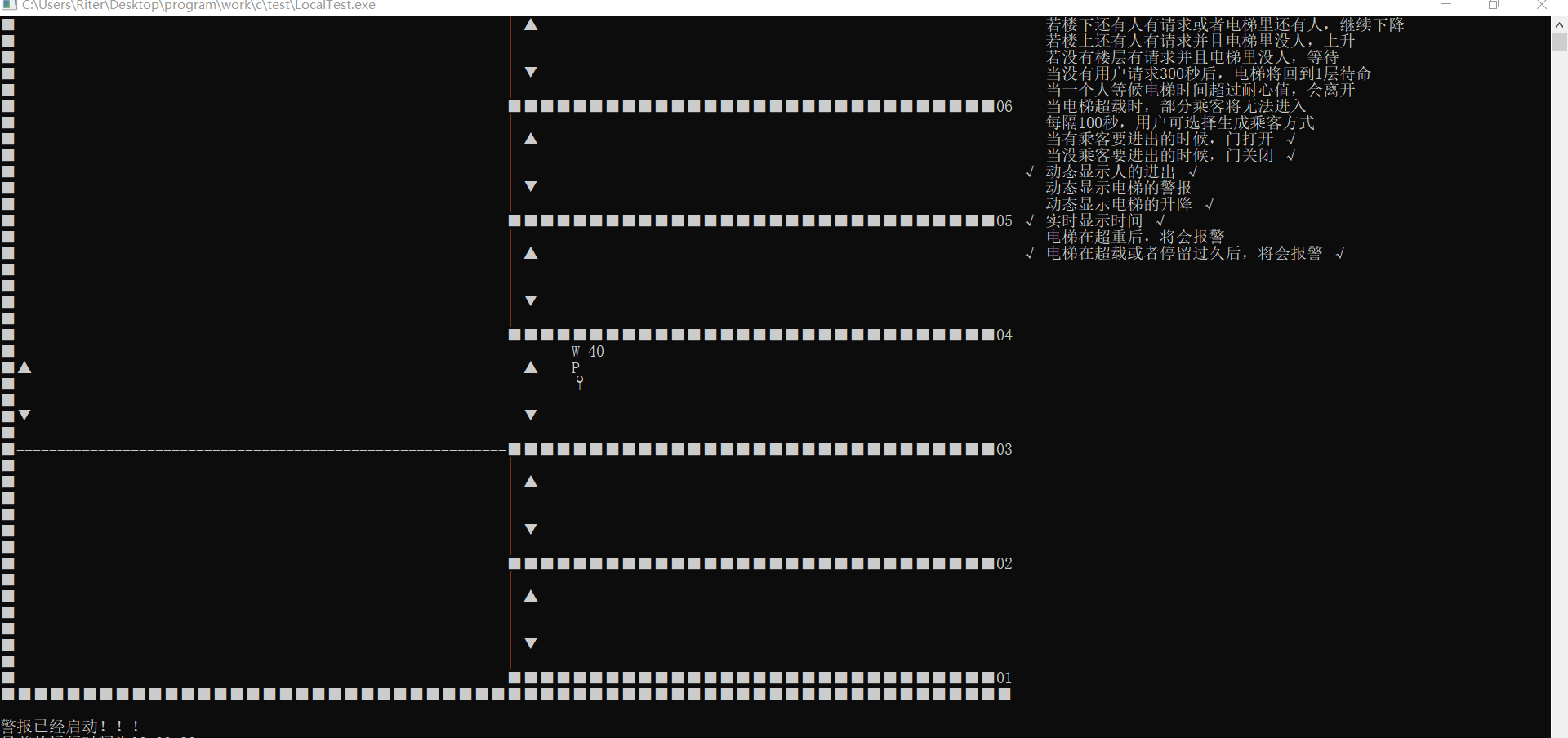




3)结论：测试通过

1. 电梯在超载或者长时间停留在除1层外的某层后，将会报警

2)实测结果



3)结论：测试通过

**六 ． 附录**

LocalTest.c//主函数

Tool.h//工具函数和工具类

Structure.h//数据结构与结构体