电梯模拟程序实验报告

题目：编写一个电梯模拟程序
班级：16级计算机科学与技术系1班
姓名：邓胜亮
学号：PB16111487
完成日期：2017年10月25日

需求分析

1. 模拟某校五层教学楼的电梯系统．该楼有一个自动电梯．能在每层停留。五个楼层由下至上依次称为地下层、第一层、第二层、第三层和第四层，其中第一层是大楼的进出层，即是电梯的“本垒层”，电梯“空闲”时，将来到该层候命。
2. 乘客可随机地进出于任何层。对每个人来说，他有一个能容忍的最长等待时间，一旦等候电梯时间过长，他将放弃。
3. 模拟时钟从0开始，时间单位为0.1秒。人和电梯的各种动作均要耗费一定的时间单位（简记为t），比如：
    - 有人进出时，电梯每隔4秒测试一次，若无人进出，则关门
    - 关门和开门各需要2t
    - 每个人进出电梯均需要25t
    - 如果电梯在某层静止时间超过300t，则驶回1层候命
4. 按时序显示系统状态的变化过程：发生的全部人和电梯的动作序列。

概要设计

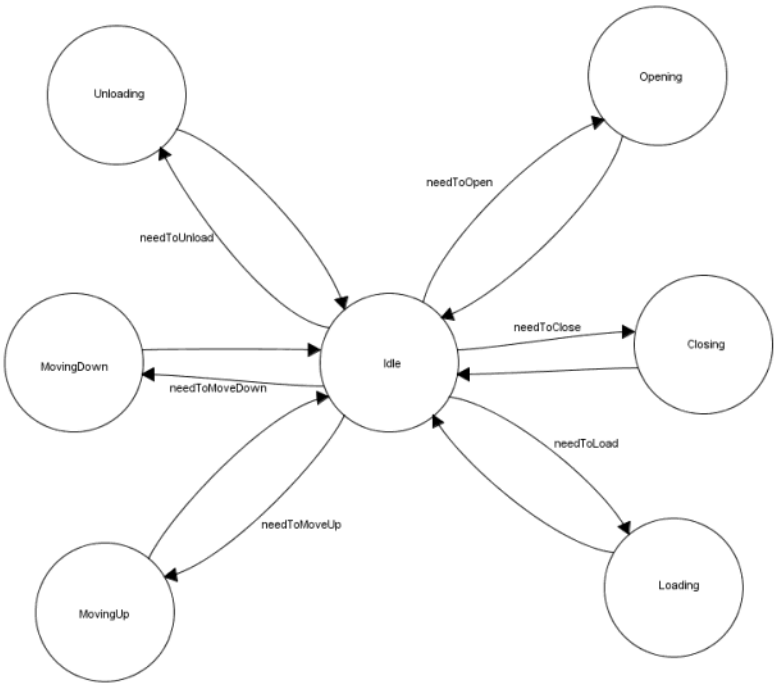
1. 该系统中的所有对象及其属性
    - 电梯
        - 所在楼层
        - 状态
        - 所载的人
    - 人
        - 出现时间
        - 起始楼层
        - 目标楼层
        - 最大等待时间
    - 楼层
        - 在该楼层排队的人
2. 因模拟需要附加的对象
    - 时钟
    - 监视器

详细设计

1. 电梯
    - 对每层楼维护一个双向链表，存储电梯中所有需要去该层楼的人
    - 对每层楼维护一个按钮列表，记录每层楼中电梯按钮的状态
    - 记录自身所在楼层以及每个状态的完成度
    - 实现为一个有限状态自动机，所有状态如下
        - Idle
            - 空闲状态
        - Loading

        - 正在等待人上电梯
    - Unloading
        - 正在等待人下电梯
    - Opening
        - 正在开门
    - Closing
        - 正在关门
    - MovingUp
        - 正在上行
    - MovingDown
        - 正在下行

    示意图如下：



    状态转移的具体规则以及优先级在这里不赘述。

2. 人
    - 在每个模拟时间单位内以一定概率出现
    - 在被创建时即设定其要到达的楼层以及最大等待时间
3. 楼层
4. 每个模拟时间单位的所有动作
    a. 随机生成一个人，将其加入所在楼层的队列（实现为双向链表）中
    b. 使电梯发生动作
    c. 判断是否有人要下电梯，如果有的话“等待”其下电梯，然后将其从系统中删除，并记录该事件
    d. 判断是否有人要上电梯，如果有的话将其从该楼层队列移动到电梯内队列中
5. 将所有被模拟的对象存储在Simulation类中，该类向外提供接口展示模拟的当前情况，包括：
    - 各层楼的人数
    - 电梯内的人数
    - 电梯所在位置
6. 使用Qt实现图形界面，用到的组件有且不限于：
    - QGraphicsScene
      用于绘制模拟图像
    - QGraphicsView

用于显示图像

- Qlabel
  用于显示界面右边的项目名称
- QLineEdit
  用于显示界面右边的具体数值
- Qtimer
  作为时钟，每隔timeGranu秒驱动模拟器完成一个时间单位的动作

7. 由于链表是该实验的核心内容，并且考虑到C++较高的表现力，在这里不写伪代码，只摘要部分头文件如下
   - personlist.h

```
#ifndef PERSONLIST_H
#define PERSONLIST_H
#include "person.h"
class PersonList
{
public:
    PersonList();
    void append(Person *p); // 在队尾添加一个人
    bool isEmpty(); // 判断队是否为空
    Person *pop_head(); // 返回队首并将其从队中删除
    int getLength(); // 得到队伍长度
    Person * getHead(); // 得到队首指针
    void remove(Person *p); // 从队中删除p
private:
    int length; // 队长度
    Person head, *tail; // 头、尾
};
#endif // PERSONLIST_H
```

   - person.h

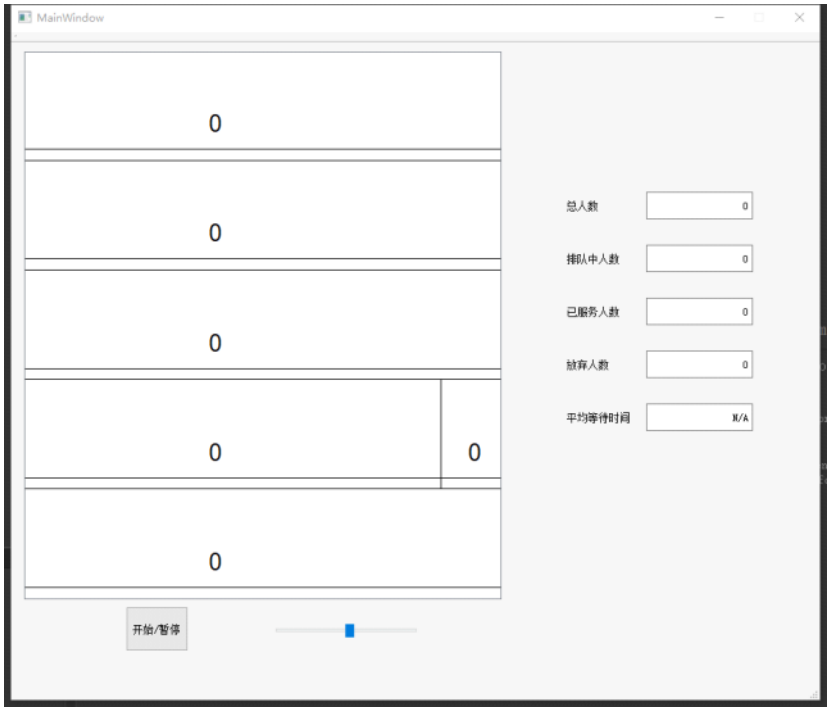
```
#ifndef PERSON_H
#define PERSON_H
class Person
{
public:
    Person(int time, int src, int dst, int pacience,
           Person *prev = nullptr, Person *next = nullptr);
    ~Person();
    int getIntension(); // 返回此人上楼还是下楼
    typedef enum {
        Queuing, // 排队中
        Resigning, // 放弃
        Moving, // 已在电梯中
        Leaving // 已下电梯
    } State; // 人的状态
    int time; // 出现时间
    int src; // 起始楼层
    int dst; // 目标楼层
    int patience; // 最大等待时间
    State state; // 当前状态
    int endTime; // 离开系统的时间
    Person *prev, *next; // 作为双向链表节点，其上一个、下一个节点的地址
};
#endif // PERSON_H
```

调试分析

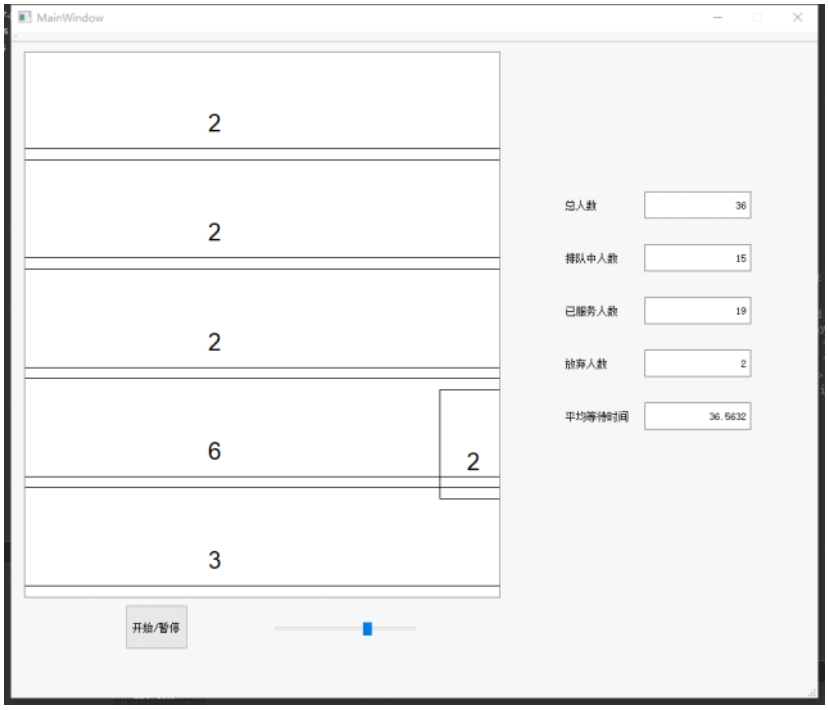
- 电梯的有限状态自动机的设计比较麻烦，状态转移条件容易写错，通过合理使用qt creator提供的debugger，成功找到了转移条件中的错误并修复

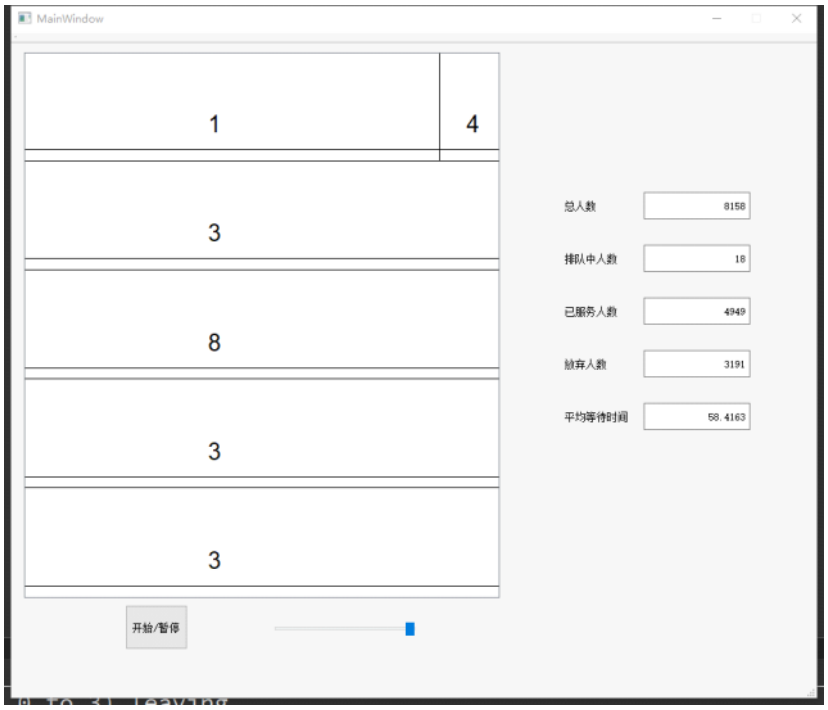
用户手册

- 程序启动后用户界面如下



- 左边的绘图区域很形象地展示了每层楼的排队中人数、电梯位置以及电梯中的人数
- 右边为统计数据，展示如下内容
  - 总人数
    即程序开始模拟至现在在系统中出现过的人的总数
  - 排队中人数
    即各楼层正在排队的人数总和
  - 已服务人数
    即已经如愿乘上电梯的人数
  - 放弃人数
    即程序开始模拟至现在放弃排队的总人数
  - 平均等待时间
    即已经如愿乘上电梯的所有人的平均等待时间
- 下方“开始/暂停”按钮可以切换模拟状态
- 按钮右边的slider可以用来调整模拟速度
  - 默认模拟速度为每0.1秒进行一步模拟
  - 向右加快模拟速度
  - 向左减慢模拟速度
- 模拟过程部分截图





- 除此之外，程序会在控制台输出电梯和所有人的动作序列，可以将其重定向到文件来记录模拟过程

附录

源程序文件名清单

- config.h

  一些全局配置，包括

  - 楼层数目
  - 繁忙程度
  - 模拟时间粒度
  - 监视器刷新频率
  - 人的最大、最小等待时间
  - 电梯完成每个动作所需时间

    - 电梯人数限制
- elevator.h、elevator.cpp
  电梯类的定义及实现
- eventgenerator.h、eventgenerator.cpp
  事件生成器的定义及实现
- person.h、person.cpp
  人的定义及实现
- personlist.h、personlist.cpp
  人的队列（双向链表）的定义及实现
- simulation.h、simulation.cpp
  对一次“模拟”的抽象
- logger.h、logger.cpp
  Logger类的定义及实现，用于统计人数、时间信息
- mainwindow.h、mainwindow.cpp
  主窗口的定义及实现
- main.h、main.cpp
  程序入口，由于使用了Qt，主函数中只是实例化MainWindow和Simulation