同济大学

# 操作系统—处理机管理

## 电梯调度

姓名：赖慧琳

学号：1652802

专业：软件工程

# 目录

# 1. 项目目的

1、学习调度算法。
2、通过实现电梯调度，体会操作系统调度过程。
3、学习特定环境下多线程编程方法。

# 2. 开发工具

## 2.1 开发环境

系统平台：windows10
Jdk: 1.8
IDE: Eclipse Java Oxygen

## 2.2 开发语言

使用 Java 语言开发。Java 编程语言的风格十分接近 C 语言、C++语言。它继承了 C++语言面向对象技术的核心，舍弃了 C 语言中容易引起错误的指针（以引用取代）、多重继承（以接口取代）等特性，增加了垃圾回收器功能用于回收不再被引用的对象所占据的内存空间。

此外，Java 语言是原生支持多线程的。通常有两种方法来创建线程：一，使用型构为 Thread(Runnable)的构造子将一个实现了 Runnable 接口的对象包装成一个线程，其二，从 Thread 类派生出子类并重写 run 方法，使用该子类创建的对象即为线程。run 方法中包含了线程所要运行的代码。Java 语言支持多个线程的同时执行，并提供多线程之间的同步机制。

# 3. 项目需求

## 3.1　基本任务

某一层楼 20 层，有五部互联的电梯。基于线程思想，编写一个电梯调度程序。

## 3.2 功能描述

1、每个电梯里面设置必要功能键：如数字键、关门键、开门键、报警键、当前电梯的楼层数、上升及下降状态等。
2、 每层楼的每部电梯门口，应该有上行和下行按钮和当前电梯状态的数码显示器。
3、五部电梯门口的按钮是互联结的，即当一个电梯按钮按下去时，其他电梯的相应按钮也就同时点亮，表示也按下去了。
4、所有电梯初始状态都在第一层。每个电梯如果在它的上层或者下层没有相应请求情况下，则应该在原地保持不动。
5、调度算法是根据所有电梯的当前状态，寻找距离当前楼层最近的且和请求同方向的电梯予以响应。

# 4. 项目设计

## 4.1 算法设计

每部电梯为一个单独的线程，对于电梯内部的数字键按钮、开门键、关门键和报警键会给予响应。电梯运行是通过判断目标楼层和当前楼层的大小关系来实现的。

① 当激活了电梯内部的数字键时，即给电梯设置一个目标楼层。若电梯正在上行，则已选择的最高楼层为目标楼层；若电梯正在下降，则已选择的最低楼层为目标楼层；若电梯是静止的，则最开始选择的楼层数决定了电梯的运行方向。

② 对于电梯外部的上下键按钮的响应，在暂停和正在上行的电梯中寻找距离楼层最近的一部来响应向上键。相反，在暂停和正在下降的电梯中寻找距离楼层最近的一部来响应向下键。响应的方式是在算法中假设电梯内部对应的楼层数字键被激活，从而执行①。

③ 电梯的移动通过区块的颜色变化来表示，利用了 java 线程的一些函数，例如线程休眠等，设置电梯停靠的时间。

④ 当电梯向上运行时，不接受低于电梯当前楼层的数字按钮响应。当电梯向下运行时，不接受高于电梯当前楼层的数字按钮响应。

## 4.2 类的设计

除了主线程函数 main 所处的类 MainRun 外，主要是 MainView 和 EleThread 两个类，还有一些监听事件的辅助类。为了异步调用，这两个类都实现了 Runnable 接口，重新定义了 run 方法。

MainView: 包括了界面的总体布局，分为六大块（五部电梯和一个楼层面板）。设计了楼层面板的布局和响应外部方向键的函数。

主要属性：totFloor 楼层总数、totList 电梯总数、eleThread 电梯线程、floorPanel 楼层面板、floorButton 楼层号、upButton 向上键、downButton 向下键.

主要方法：run()线程调用、searchListdown(int)寻找响应向下键的电梯、searchListup(int)寻找响应向上键的电梯

包含的类：moveButtonAction：继承自 MouseAdapter，实现了 MouseListener 接口，用于注册电梯外部上下行键的监听器。

EleThread: 代表着一部电梯以一个线程的方式响应事件。设计了电梯面板的布局和响应内部按钮的函数。

主要属性：direct 移动方向、curFloor 当前楼层、goaFloor 目标楼层、runStart 电梯运行、numbButton 数字键、floorBlock 电梯移动块、open 开门键、close 关门键、warning 报警键、showlistState 显示电梯状态、showcurFloor 显示当前楼层、

主要方法：run()线程调用、listup()电梯上行、listdown()电梯下降、setnumbState()重清数字键状态、setList(int)电梯响应上下行键

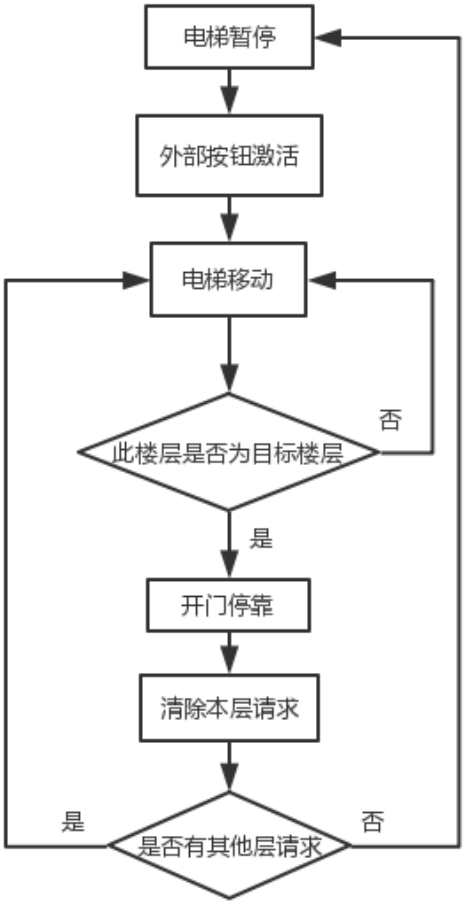
包含的类：numbButtonAction：继承自 MouseAdapter，实现了 MouseListener 接口，用于注册电梯内数字键的监听器。

opendoorThread：继承自 Thread，实现开门键的功能。

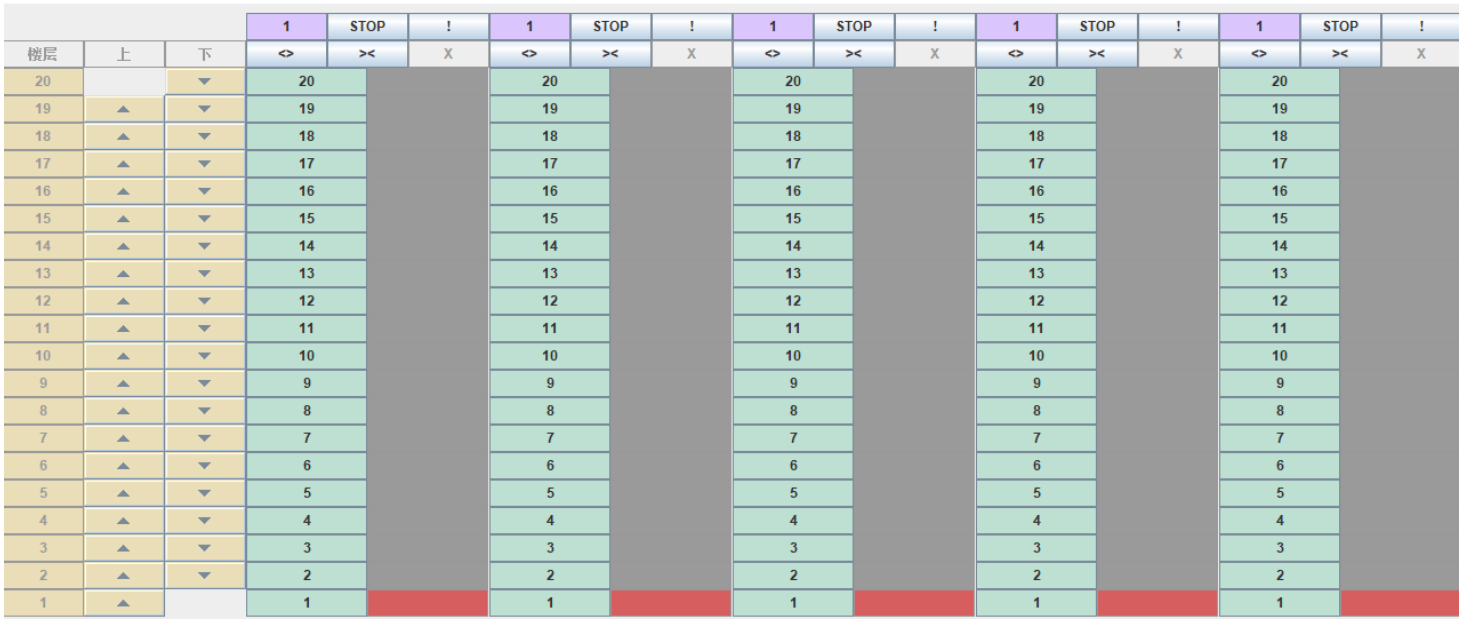
closedoorThread：继承自 Thread，实现关门键的功能。

ActionListener：使用匿名类，注册开关门键、警报键和取消警报键的监听器

## 4.3 流程图



## 4.4 界面设计




界面总共分为六大区。

最左边区表示每层楼的每部电梯门口，有楼层号、上行键和下行键，最底层无下行键，最高层无上行键。

其余五部分为五部电梯区，每区左边是楼层数字键，可以被点击，右边模拟电梯上下移动。最上面两排是电梯内部的一些其他按钮和显示。第一行紫色框动态显示当前所在楼层，中间框会显示电梯状态，比如暂停、上升、下降、开门、关门等。“!”框表示警报键。按下会产生“Alert!”提示，且电梯停止运行。第二行前两个开门键和关门键，只有电梯在暂停状态，开关门键才会有效。“X”框表示解除警报，电梯正常运行。

# 5. 实现与演示

## 5.1 核心代码

1. 寻找响应外部向上键的电梯：

```
int elevNo = -1;
int minDist = totFloor;
for(int i = 0; i < totList; i++) {
    //只有暂停的和正在下降的且无报警的电梯能响应向下键
if((eleThread[i].isSTOP() || (eleThread[i].isDOWN() &&
eleThread[i].getcurFloor() >= floor)) && !eleThread[i].getwarnFlag())
{       int dist = Math.abs(floor - eleThread[i].getcurFloor());
        if(dist < minDist) {
            elevNo = i;
            minDist = dist;
        }
    }
}
```

2. 寻找响应外部向下键的电梯：

```
int elevNo = -1;
int minDist = totFloor;
for(int i = 0; i < totList; i++) {
    //只有暂停的和正在上升的且无报警的电梯能响应向上键
    if((eleThread[i].isSTOP() || (eleThread[i].isUP() &&
  eleThread[i].getcurFloor() <= floor))&& !eleThread[i].getwarnFlag())
  {     int dist = Math.abs(floor - eleThread[i].getcurFloor());
        if(dist < minDist) {
            elevNo = i;
            minDist = dist;
        }
    }
}
```

4. 目标楼层的设定：

```
 if(direct == 0) {
   goaFloor = i;
}
 //若电梯正在上行，则已选择的最高楼层为目标楼层
 if(direct == 1) {
   goaFloor = 0;
   for (int j = totFloor - 1; j >= 0; j--) {
       if (numbState[j]) {
           goaFloor = j;
```

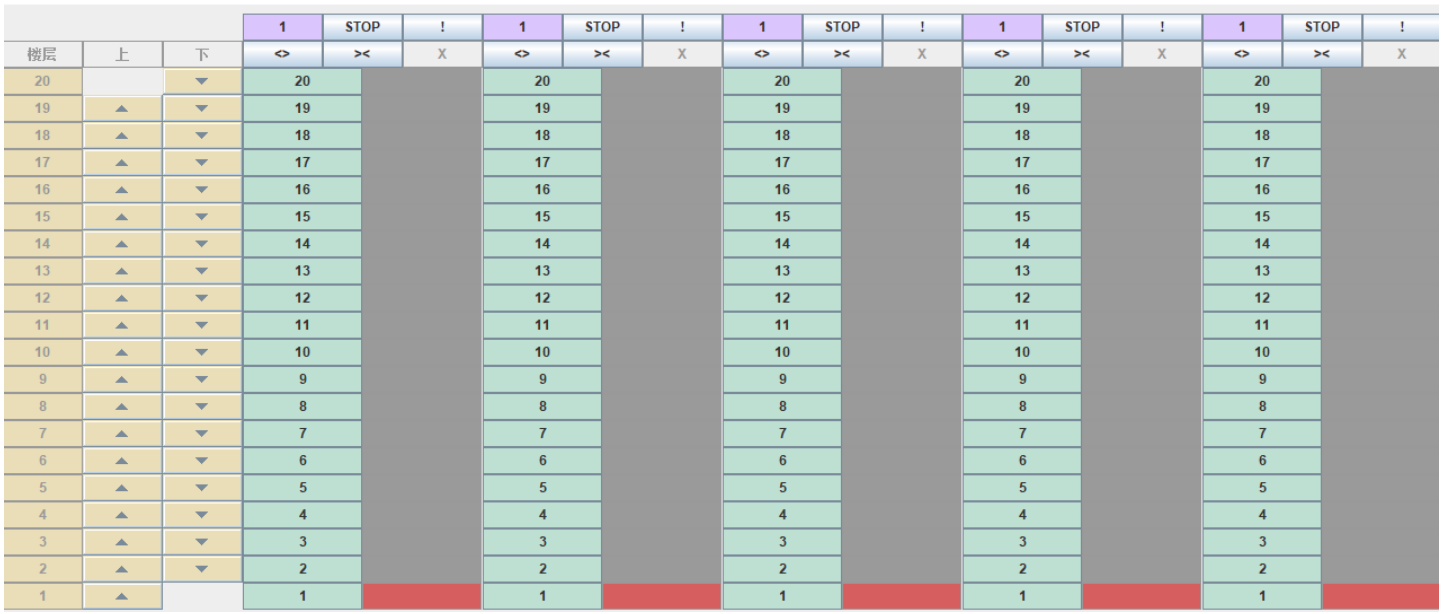
```
                    break;
                }
            }
        }
          //若电梯正在下降，则已选择的最低楼层为目标楼层。
           if(direct == -1) {
             goaFloor = 0;
             for (int j = 0; j < totFloor; j++) {
                if (numbState[j]) {
                    goaFloor = j;
                    break;
                    }
                }
            }
```

5. 电梯响应上下键：

```
        if(direct == 0) {
            goaFloor = floor;
            numbState[floor] = true;
            if(curFloor > goaFloor) {
                direct = -1; //下降
                ……
            }
            else if(curFloor < goaFloor) {
                direct = 1; //上升
                ……
            }
        }
        if((direct == 1 && floor > goaFloor) || (direct == -1 && floor < goaFloor))
        {
            goaFloor = floor;
            numbState[floor] = true;
             ……
        }
        else if((direct==1 && floor > curFloor)||(direct==-1 &&floor < curFloor))
          {
            numbState[floor] = true;
             ……
          }
```

## 5.2 演示说明

双击 电梯调度.jar 可直接运行，因为程序是用 Java 语言编译的，需安装 jdk 1.8。




1、 初始界面如上，所有电梯都停在第一层（砖红色表示），处于未运行状态。电梯内按键除了解除警报键在按下警报键后才有效，其余均有效。电梯门口上下行键都有效，最底层无下行键，最高层无上行键。第一行的紫色框动态显示对应电梯所处于的楼层号。


2、 当电梯门口的上下行键被按下后，需响应的电梯内块为深紫色，电梯向该方向移动。

3、 当电梯内部数字键被按下后，呈浅橘色，电梯向该方向移动。

4、 当电梯到达上下行键的楼层时，上下行键状态复原，电梯自动开关门。

5、 当电梯到达目标楼层后，电梯内的按键状态全部复原。

6、 当电梯上行时，不响应低于当前楼层的请求；当电梯下降时，不响应高于当前楼层的请求。

7、 电梯移动速度为 600ms 一层。


8、当电梯到达要停靠的楼层时，会开门，此时电梯块设置为黄色，900ms 后关门。可根据需要暂停或继续运行。

9、按下警报键，则如上图中第二部电梯所示，显示红色框和“Alert!”，电梯不会再工作。同时，解除警报键可有效，按下接触警报键后电梯才可正常工作。

10、当电梯停靠时，可手动按下开门键，此时可手动关门或者 900ms 后会自动关门。

# 6. 心得体会

此次项目作业是使用 java 语言编写的，目的是更好利用线程和展示界面。因为之前基本都是用 c++语言，所以花了一些时间来熟悉 java，同时也参考了历年的项目实例，再根据自己的理解来编写项目。

首先比较容易地搭建了界面布局。然后实现电梯的基本调度功能，选择的是与电梯运行方向不矛盾中的距离楼层最近的一部电梯作为响应者，且要保证所有请求都被响应。在完成基本功能后，为了更好地展示界面中的反馈，增加了不同颜色表示不同含义，例如当电梯门口的上下行键被按下后，需响应的电梯内块为深紫色。接着，为了加强写实度，增添了开门、关门、和警报等的功能，相关按钮能实现其基础功能。

经过多次调试运行来修正代码，对于一个 java 小项目有了一个简单的了解，得到了很多有意义的经验。但这个电梯调度程序也仍然不够完善，例如程序中没有表示电梯的搭载人数。此外，程序如果能实现记载电梯调度日志，方便统计用户调度电梯的情况，就可以优化调度算法。