项目说明文档

操作系统课程设计

——电梯调度

作 者 姓 名： 刘冬阳

学 号： 1753575

指 导 教 师： 王冬青

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

目 录

[1分析 1](#_Toc8485525)

[1.1 问题重述 1](#_Toc8485526)

[1.2 功能分析 1](#_Toc8485527)

[2 设计 1](#_Toc8485528)

[2.1 开发环境 1](#_Toc8485529)

[2.2 按键设计 1](#_Toc8485530)

[2.3 指示信息设计 2](#_Toc8485531)

[2.4 电梯标志设计 3](#_Toc8485532)

[2.5 线程设计 3](#_Toc8485533)

[2.6 信号量设计 3](#_Toc8485534)

[2.7 类设计 3](#_Toc8485535)

[3 实现 4](#_Toc8485536)

[3.1程序主线程逻辑 4](#_Toc8485537)

[3.1.1 活动图 4](#_Toc8485538)

[3.1.2 描述 4](#_Toc8485539)

[3.2 电梯外部调度逻辑 6](#_Toc8485540)

[3.2.1 活动图： 6](#_Toc8485541)

[3.2.2 描述 6](#_Toc8485542)

[3.3 电梯线程运行逻辑 8](#_Toc8485543)

[3.3.1 活动图 8](#_Toc8485544)

[3.3.2 描述 8](#_Toc8485545)

[3.4 开门过程 13](#_Toc8485546)

[3.4.1 活动图 13](#_Toc8485547)

[3.4.2 描述： 14](#_Toc8485548)

[4 程序展示 15](#_Toc8485549)

[4.1 程序初始化 15](#_Toc8485550)

[4.2 程序运行中 16](#_Toc8485551)

# 1分析

## 问题重述

某一栋楼有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。每个电梯里面设置必要功能键：如数字键、关门键、开门键、上行键、下行键、报警键、当前电梯的楼层数、上升及下降状态等。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。

## 1.2 功能分析

1.实现图形界面。

2.实现面向用户操作的电梯调度算法，在合理、可行的基础上尽量高效。

3.运用多线程编程。

# 2 设计

## 2.1 开发环境

本程序使用c++语言配合Qt框架构建。Qt是一个1991年由Qt Company开发的跨平台C++图形用户界面应用程序开发框架。它是面向对象的框架，使用特殊的代码生成扩展以及一些宏。Qt很容易扩展，并且允许真正的组件编程。

## 2.2 按键设计

本程序设计的按键包括电梯外部的上下键，电梯内部的数字键、开关门键和报警键。现对按键功能做简要介绍：

上下键：

电梯外部的上下键有两个功能：

1. 首先，在一般情况下，上下键用来召唤一部电梯。以上键为例，当上键被按下后，其会保持点亮状态。电梯调度系统会指派最近的空闲电梯前去接客。如果没有空闲电梯，则请求会被压入等待队列，待有电梯空闲出来后接受该任务。当被专门指派到该楼层的电梯，或者一部本身使命与该层请求不冲突的电梯到达该层后，电梯会开门，同时上键的点亮状态会结束（恢复正常状态）。下键的情况相同。
2. 特别的，当一部电梯正好停靠某层且处于开门状态，则该层与此电梯对应上下键中，与电梯运行状态相同的按键将和电梯内的开门键发挥同样的作用，即，按下后将阻断电梯的关门操作，保证电梯门打开直到按钮被松开。

数字键：

电梯内部数字键，用于电梯内部用户对电梯的操作。如果电梯内某一层的按钮被点亮，则电梯会在该层停靠。

开关键：

开关键在电梯移动中被置为无效，只有当电梯静止在某一特定楼层时才有作用。关门键可以缩短电梯门保持打开的时间；开门键按下后将阻断电梯的关门操作，保证电梯门打开直到按钮被松开。

报警键：

由于报警键的设计考虑因素较多，已经超出了本模拟程序所能模拟的范围，故本程序对报警键进行了简化，电梯中的每一层均会被按下，以期待电梯内被困人员能够得到开门机会。

## 2.3 指示信息设计

本程序加入了电梯楼层显示器和电梯状态指示灯。

电梯楼层显示器：

在现实中，每个电梯在每一层都对应有一个楼层显示器，并且在电梯内部也应设置楼层显示器；在本程序中，由于空间限制，每部电梯仅设置一个楼层显示器。楼层显示器将实时显示电梯所在楼层。

电梯状态指示灯：

与楼层显示器相同，每部电梯在本程序中仅设置一个电梯状态指示灯。电梯状态指示灯在电梯运行时实时显示电梯运行状态（上、下或空闲）。值得注意的是，电梯静止并不代表电梯空闲：电梯在开门后仍记录着原有的方向信息，在关门后将优先保持原来的方向继续运行，只有当原方向没有任务后，电梯才会根据反方向是否有任务决定切换方向或原地待命（置为空闲）。

## 2.4 电梯标志设计

本程序使用进度条表示电梯门，当进度条置为满时电梯门为关闭状态；当进度条为空时，电梯门为完全打开状态。

## 2.5 线程设计

本程序涉及到六个线程，其中包括五个电梯线程和一个主线程（调度线程）。主线程用于调度五部电梯，并执行一切和ui有关的操作（Qt要求ui相关操作只能由主线程执行）。五部电梯负责处理来自电梯内部按钮的请求、接收并服从来自主线程的调度并通过计算更新电梯的各状态变量。

## 2.6 信号量设计

本程序设计了三类信号量，分别用来保证任务等待对列、每部电梯的自身状态信息（运行方向、是否空闲等）和每部电梯的开关门状态（是否保持开门）在特定时刻只有一个线程访问。

## 2.7 类设计

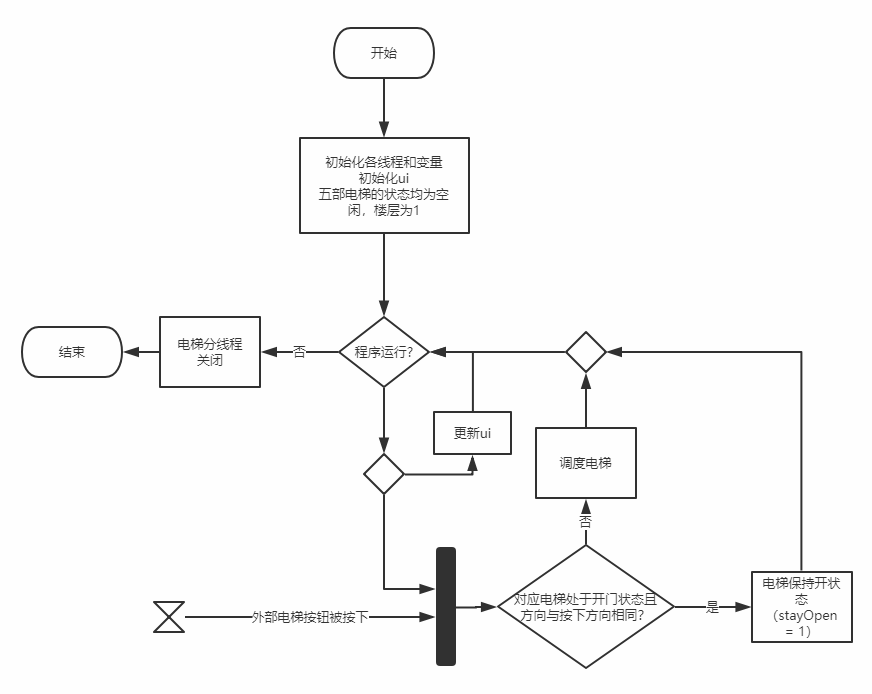
本程序使用面向对象方式开发，所设计的类可以分为以下四个类型：

1. 总控类：dispatcher（单例模式）
2. 电梯线程类：LiftThread
3. 电梯类：Lift
4. 封装的ui类：InButton，OutButton，LabelButton等

# 3 实现

## 3.1程序主线程逻辑

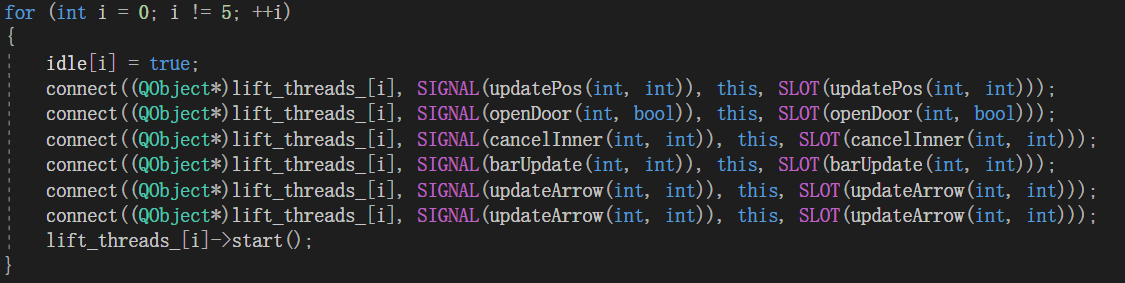
### 3.1.1 活动图



### 3.1.2 描述

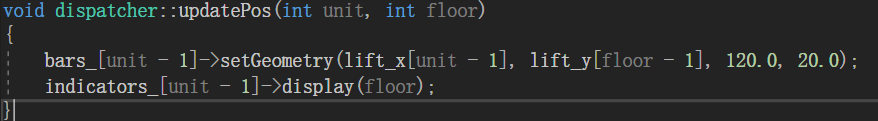
程序开始后主线程初始化，首先对各状态变量、全局变量进行初始化，并对ui界面中的内容进行布置。

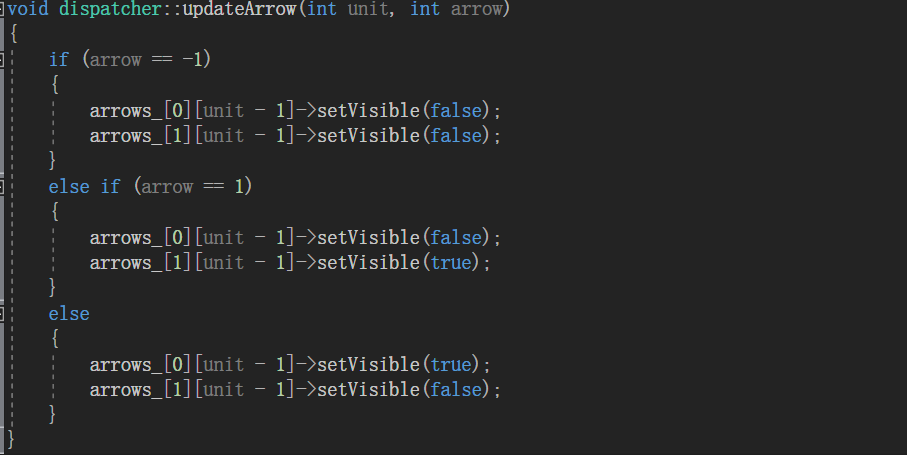
之后，主线程的工作可以被分为两部分：更新ui界面和处理电梯外部的调度请求。这两种功能的实现都借助Qt的信号槽机制，即，当ui界面中的按钮被点击后或电梯线程发送信号后，主线程会调用先前约定好的函数，对这些请求做出相应。



信号——槽连接设置

每个电梯子线程按周期运行，在每个周期的末尾，电梯线程会向主线程发送对应电梯的位置等状态信息。主线程在接收这些信息后，会据此对ui进行更新。



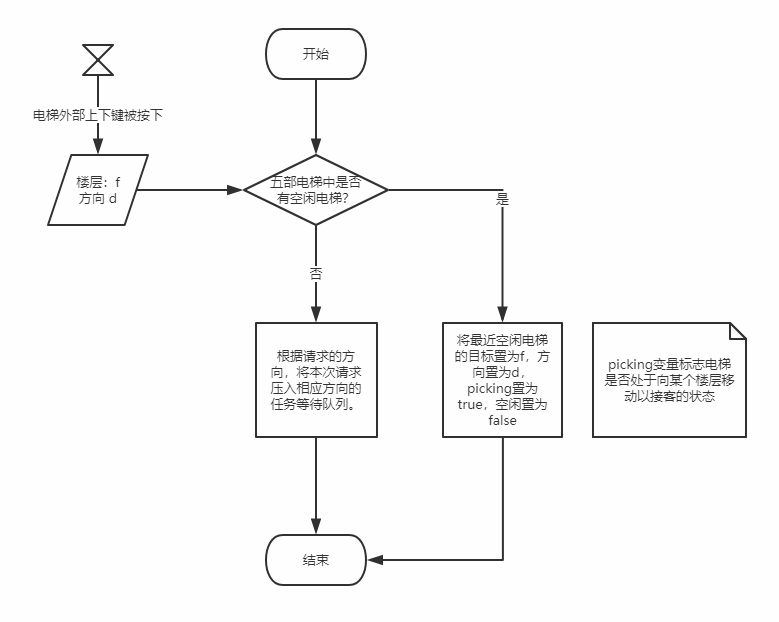


例：更新ui的函数

除了控制ui，主线程还负责处理电梯的外部调度，即，控制电梯对电梯外上下键的响应。

## 3.2 电梯外部调度逻辑

### 3.2.1 活动图：

电梯外部调度逻辑

### 3.2.2 描述

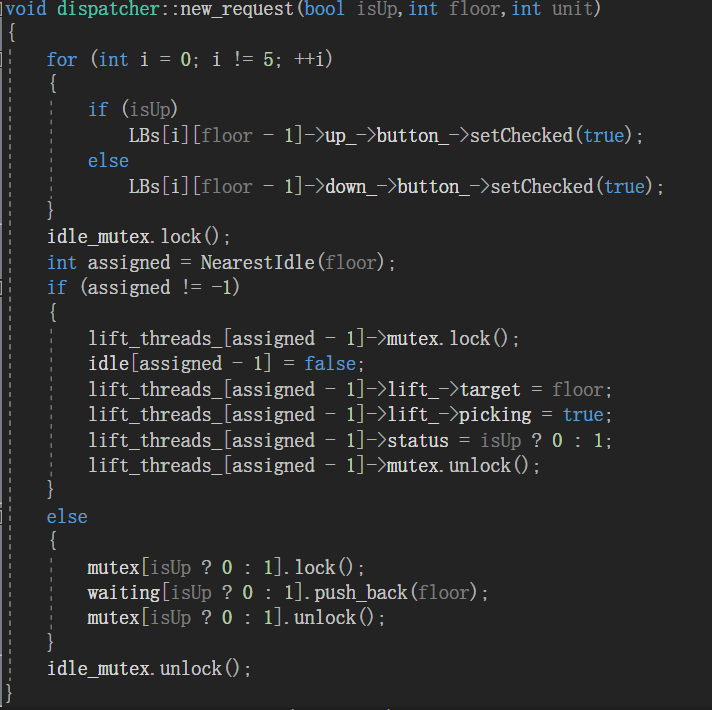
当电梯外部按钮的上或下键被按下，且确认其作用为申请调用电梯而非保持电梯门开后，主线程会执行电梯外部调度逻辑。

首先，主线程将依次扫描五部电梯的空闲状态变量idle，以找到最近的空闲电梯。由于电梯线程和主线程可能就idle变量发生写-写冲突，且其后果会导致一个请求不被完成而直接消除，我们使用一个信号量来保证同一时间只有一个线程对该变量进行访问。

找到最近的空闲电梯后，主线程将该电梯的状态变量，包括picking（指示电梯是否处于向某个楼层移动以接客的状态），idle（指示电梯是否空闲），target（当picking==true时，指示电梯的接客目标）等。由于电梯线程会在循环中检查这些状态量并依此决定操作，位于主线程的调度器在修改完这些状态变量后便完成了对一次请求的调度。

如果五部电梯均不是空闲状态，则本次请求的信息会被加入等待队列。当有电梯完成现有使命后，其会检查等待队列，并执行队头请求；只有当电梯现有任务完成且请求队列为空时，电梯才会重新置为空闲状态。

特别需要指出的是，如果其余电梯在执行自身任务时，正好路过一个具有等待请求的楼层，且该请求的方向与电梯自身任务不冲突，则此电梯会开门响应请求，并将等待任务从队列中消去。

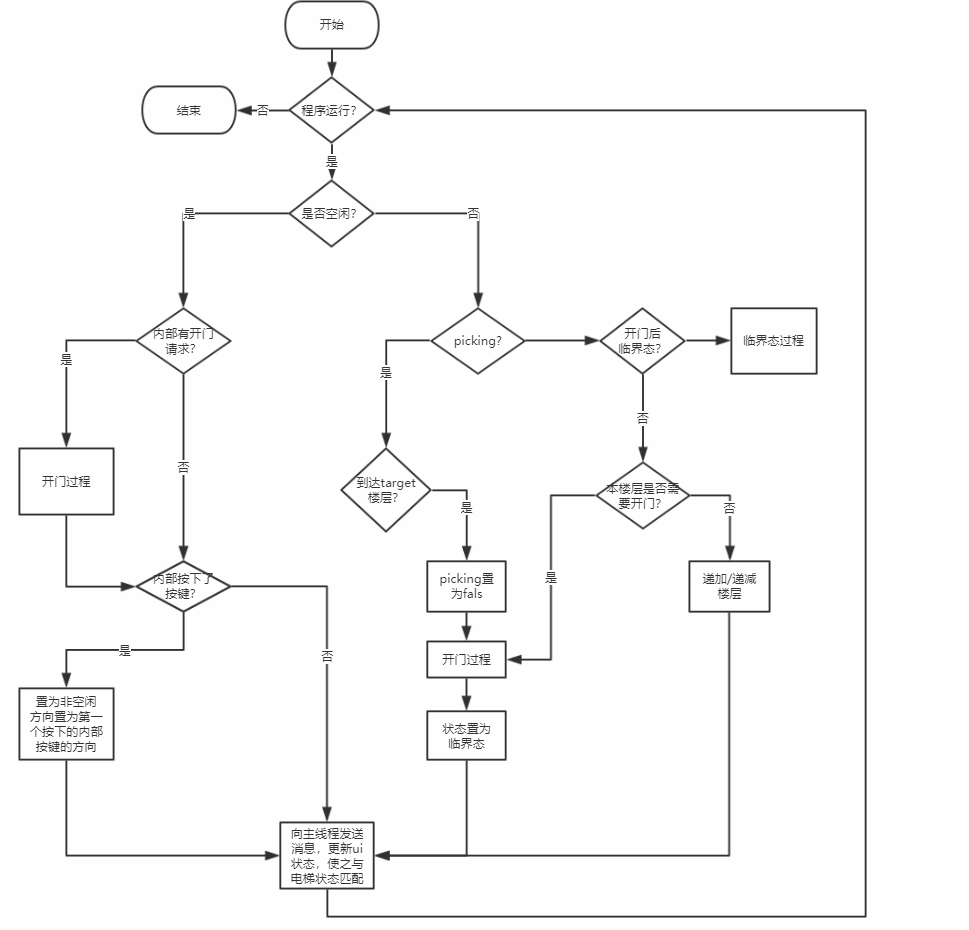


电梯外部调度逻辑

上图是此调度逻辑的实现。其中idle\_mutex是信号量。

## 3.3 电梯线程运行逻辑

### 3.3.1 活动图



### 3.3.2 描述

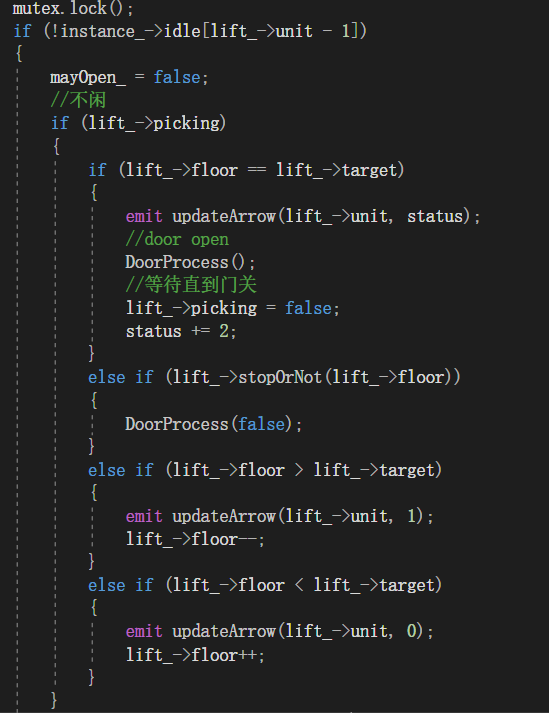
与主线程只有在接收到信号后才会主动调用函数不同，电梯线程始终处于循环之中，在每个循环的开始检查对应电梯的状态，根据电梯的状态决定下一步操作。在一个电梯周期内，能够执行的操作有执行一次开-关门流程和电梯移动一层，同时还可以对电梯状态做出改变，从而使得下一个周期开始时电梯按照预期的行为运行。

进入新的电梯周期后，如果电梯线程发现自身为空闲，则说明自从上次电梯完成自身任务到现在，电梯没有新的来自内部或外部的任务。此时，电梯会依次检查stayOpen变量和电梯内部的数字键，前者如果为真，代表电梯内部有人请求开门，电梯会执行开门过程；后者如果为真，代表电梯收到了来自内部的运行请求，此时电梯会把自身状态置为运行，并从下一个周期开始执行新的任务。如果电梯内部数字键都没有被按，则电梯将保持当前状态不变。

如果电梯并不空闲，则可分为picking状态和非picking状态。

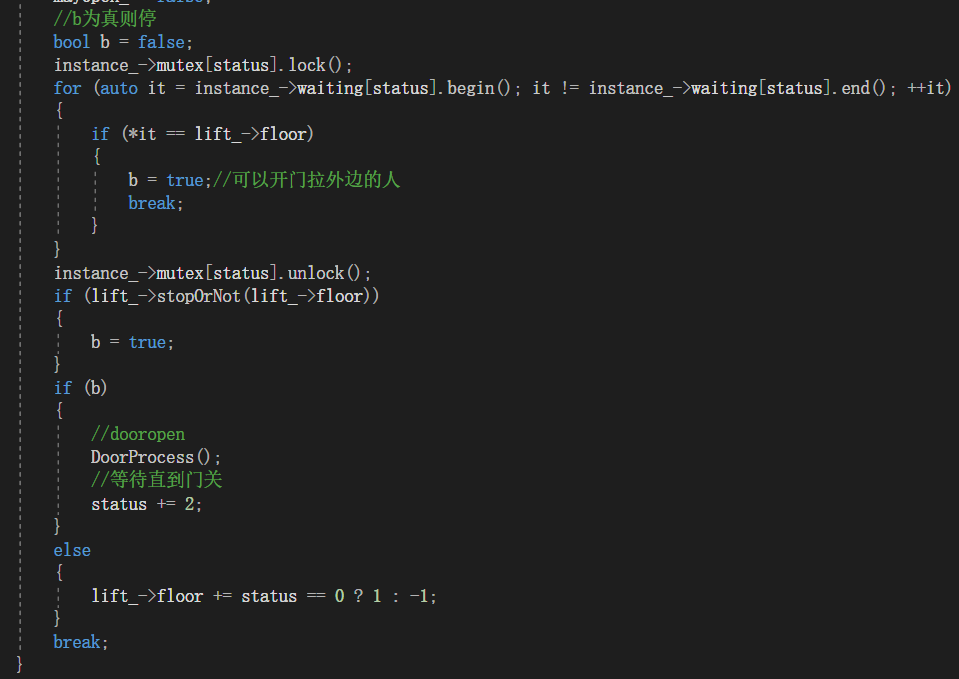
在picking状态下，如果电梯发现自己现在所在楼层为target楼层，代表电梯已经到达了调度器指派的接客位置，会将picking置为false，并调用开门过程。开门后，电梯将暂时的进入开门后的临界态。该临界态设立的目的是给电梯内部用户提供了一定的反应时间，避免电梯门关闭后，乘客还来不及按下楼层键时电梯就误以为自己没有任务并响应新的请求。

如果电梯发现自己所在楼层不是target楼层，电梯将先检查是否需要在本层开门。由于电梯具有去某一层接客的任务，因此不会响应其他顺路的外部请求，但内部请求仍将被响应。即，电梯不会因为所在楼层的上或下键被按下而在该层停下，但会因为所在楼层在内部被按下而在该层开门。如果需要开门，则会执行开门过程，否则电梯将向target逼近一层。



Picking分支

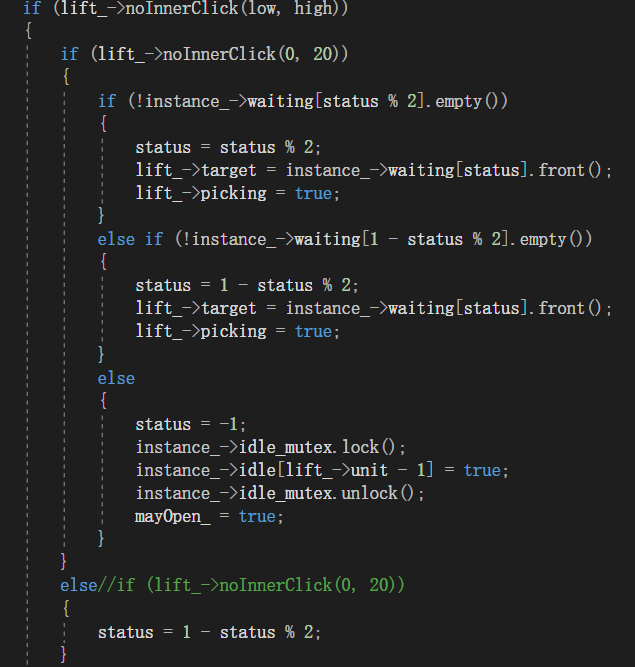
如果电梯不处于picking状态，则电梯可分为一般运行态和临界态。在一般运行态下，电梯会首先检查当前楼层是否有需要响应的内部请求或者顺路的外部请求，若有，则电梯会执行开门过程；若没有，则电梯会根据status变量的值获悉其运行的方向，并向该方向行进一层。



一般运行分支

电梯处于临界态，代表电梯刚刚完成了开门过程，此时电梯会执行临界态过程。在临界过程中，电梯需要确定自身是否已经完成了既定任务，以及是否需要响应其他任务。

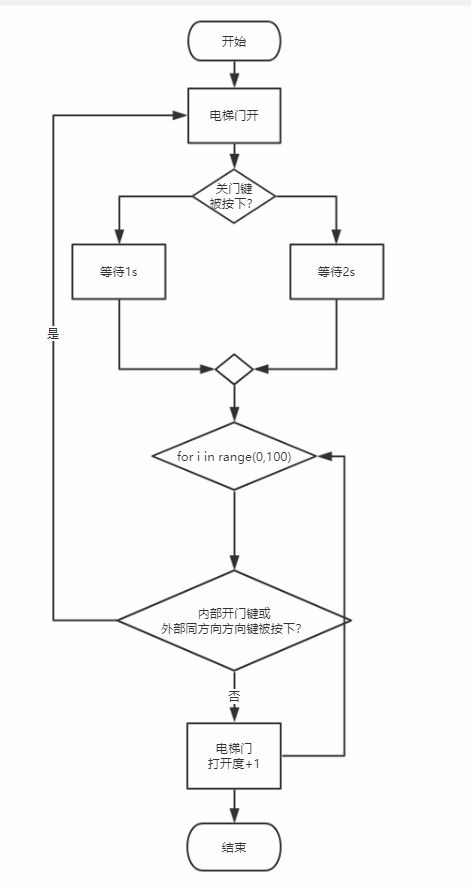
首先，电梯将检查自身内部的按键中，是否有被按下的同方向数字键（举例：如果电梯位于10层，原本方向为向下，且2，3，13，15层在内部被按下，则2，3是同方向的，13，15是异方向的）。如果有，则电梯将恢复一般运行状态；否则电梯将检查是否有被按下的异方向数字键。若存在内部被按下的异方向数字键，则电梯将改变运行方向，并进入一般运行状态；否则， 电梯将检查处于等待状态的外部调度请求。如果存在等待中的外部调度请求，则电梯会接受最早的一个；否则，电梯将进入空闲状态。



临界态分支

## 3.4 开门过程

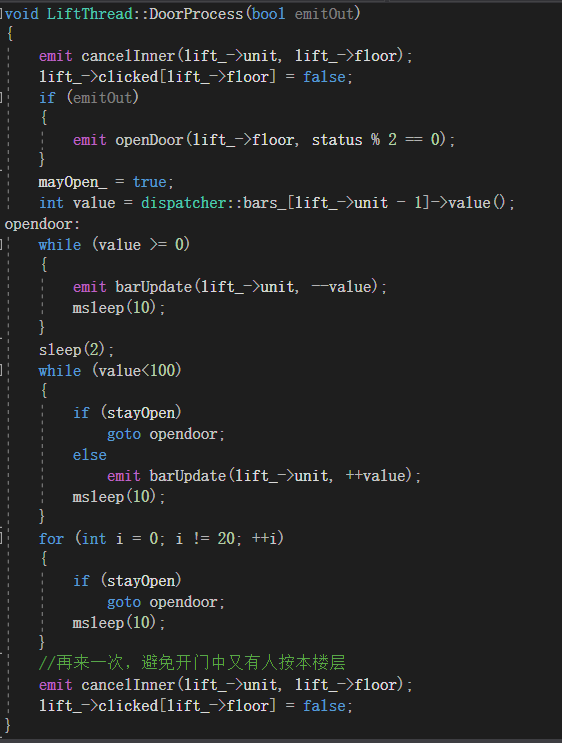
### 3.4.1 活动图



### 3.4.2 描述：

电梯的开门过程可以被分为三个阶段。首先是电梯门打开过程，该过程是不可被打断的，即，执行后电梯门一定会开到最大。接下来是等待流程，默认电梯会保持打开2s，如果关门键被按下，则会打开1s。最后是关门流程，这个过程会被打断。

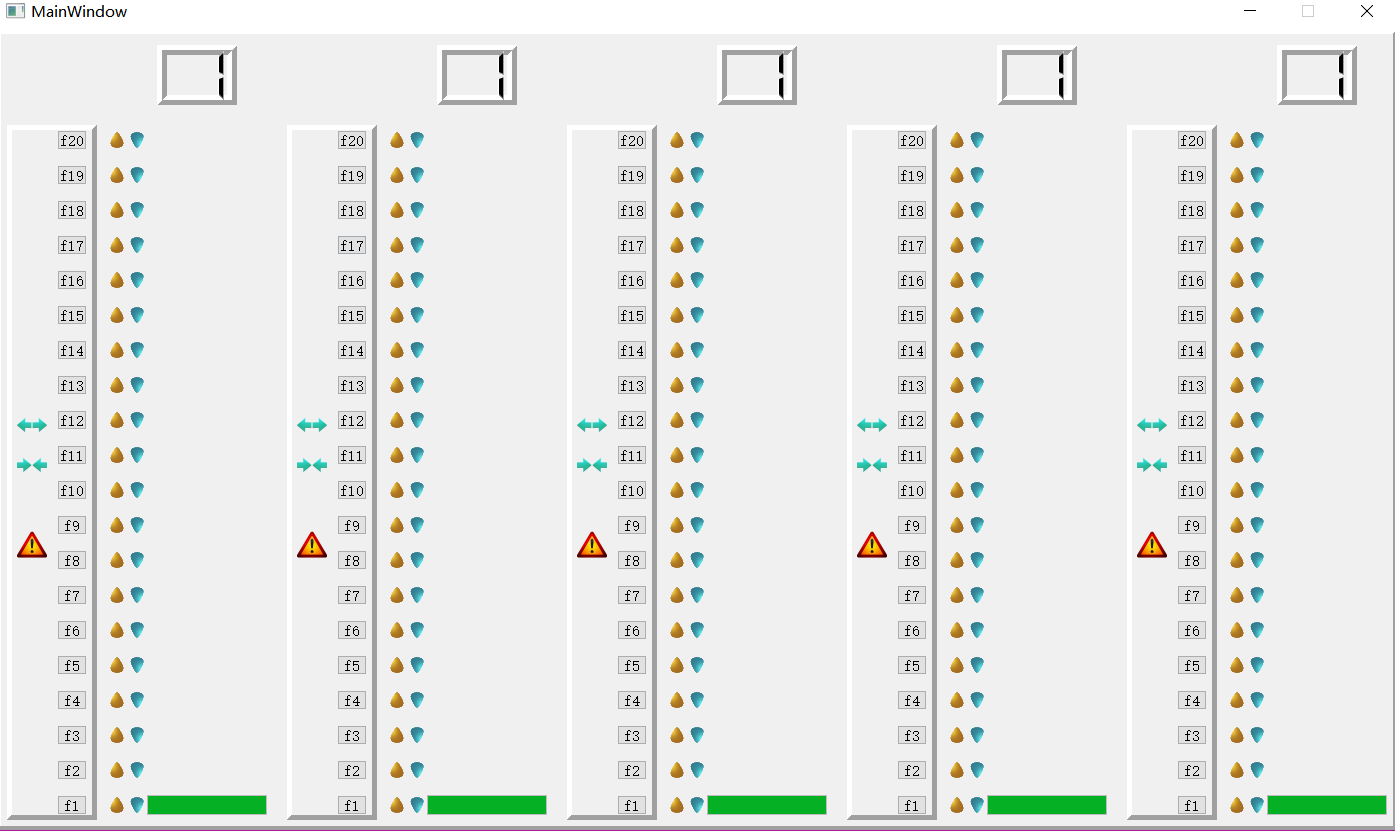
这里就第三阶段做出说明。对现实中的电梯进行分析发现，当电梯门要关闭时，通过按电梯内部的开门键和电梯外的与电梯运行方向相同的上/下键，均可以打断关门过程，从而让电梯门重新打开。本程序也据此设计了开门过程。



开门过程

# 4 程序展示

## 4.1 程序初始化



## 4.2 程序运行中

