OO-第十四次作业-正确性论证

2017/6/4

# 调度类（Dispatch）

## 1.1 抽象对象有效实现



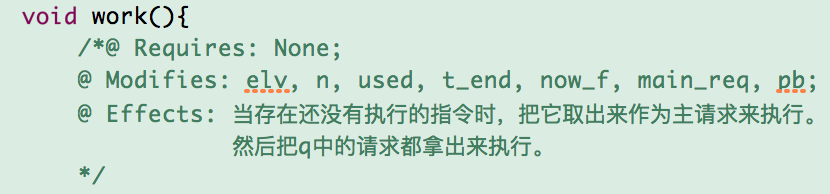
表示对象为elevator、请求的floor、type、updown、time，通过抽象函数映射为有请求的楼层、类型、上下、时间的信息，以及一个待调度的电梯。

## 对象有效性

* + 1. Elevator()是构造函数，新建一个电梯对象，调用上层的调度机初始化请求的各个数组的信息，使得都不为空，满足不变式。同样也满足父类的不变式。
    2. mymax为求两个数的较大值，out\_s为按标准格式输出请求的信息，flag\_req为判断请求i目前能够执行。这些方法都不会修改对象，因此不会影响repOK，也不会更改父类的rep.
    3. work()、exec()实现整个ALS的调度过程，但整个过程不会更改请求的相关数组，也不会让Elevator变成null，因此满足不变式，同样也满足父类的不变式。

所以，所有方法执行前和执行后的repOK都为true。综上，对该类任意对象的任意调用都不会改变其repOK为true的特性。因此该类任意对象始终保持对象有效性。

## 方法实现正确性



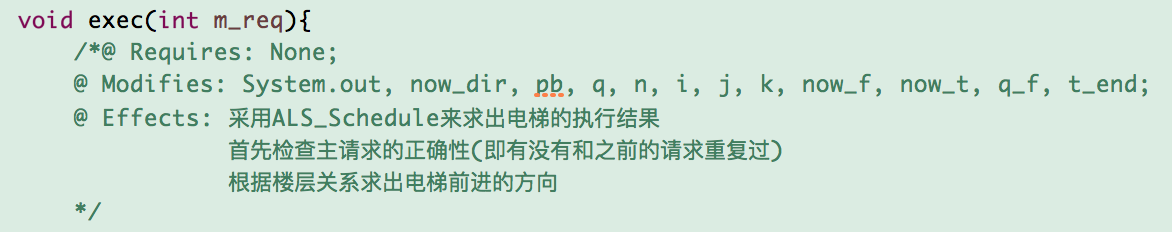
<carry out with request i> with <\exist i;0< i < n; use[i] == 0>

<carry out with request j> with <\all j; q.contains(j) && \exist i;0< i < n; use[i] == 0>

<do nothing > with <\all i;0< i < n; used[i] == 1>

方法来寻找当前的used[i]为0的请求，代表这个请求还没执行，此时我们把这个请求标记为主请求，然后去执行exec函数。然后把所有的q中的请求拿出来执行。

如果没有找到的话，就一个exec函数都不执行。



<do nothing> with <!flag\_req(m\_req)>

<set dir to 0> with <floor[m\_req] > now\_f>

<set dir to 1> with <floor[m\_req] < now\_f>

<set dir to 0> with <floor[m\_req] == now\_f>

<q.add(i)> with <\exist i;0< i < n; 主请求可以捎带i>

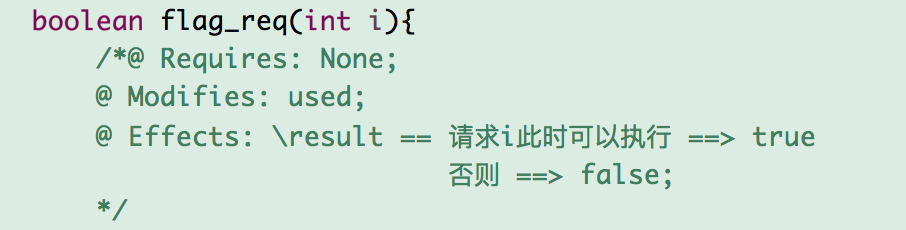
<q.poll(i)> with < elv.getFloor() == floor[q.peek().getI()]>

如果通过flag\_req判断出当前的主请求为之前的重复请求，那么什么都不执行。

根据楼层关系求出电梯前进的方向。0为向上，1位向下，2为静止。

循环判断有没有当前的主请求可以捎带的请求，如果有就增添到q中。

如果q的队首到达了我们的当前楼层，就从q里面弹出。

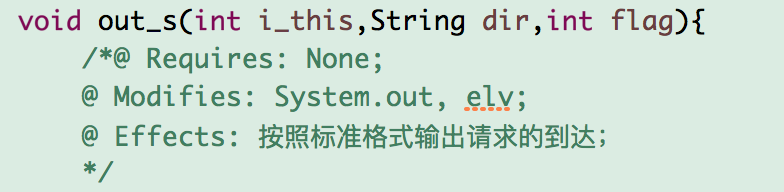


<return true> with <\exist i;0< i < n; time[i]==time[j]&&type[i]==type[j] &&floor[i]==floor[j]&&updown[i]==updown[j]>

<return false> with <\all i;0< i < n; time[i]!=time[j]||type[i]!=type[j] ||floor[i]!=floor[j]||updown[i]!=updown[j]>

如果有一个请求的信息和请求i的信息一目一样，那么返回TRUE；

否则返回FALSE。

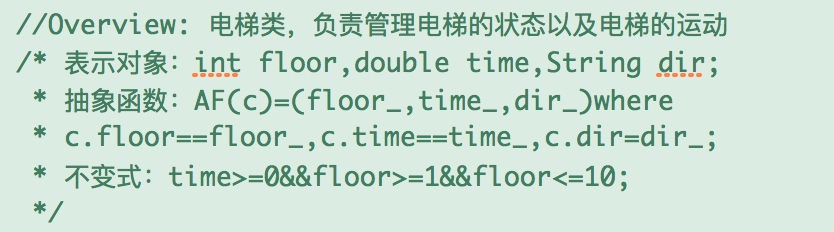


<print information of arrival in standard form> with <any>

综上所述，所有方法的实现都满足规格。从而可以推断，dispatch的实现是正确的，即满足其规格要求。

# 电梯类(Elevator)

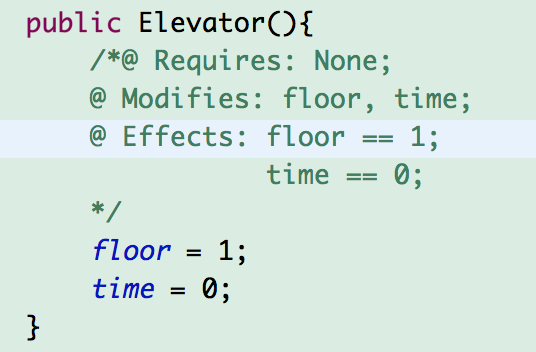
## 抽象对象有效实现



表示对象为state、floor、time，通过抽象函数映射为有当前时间、当前所在楼层、运行状态（字符串信息）的电梯。

## 对象有效性

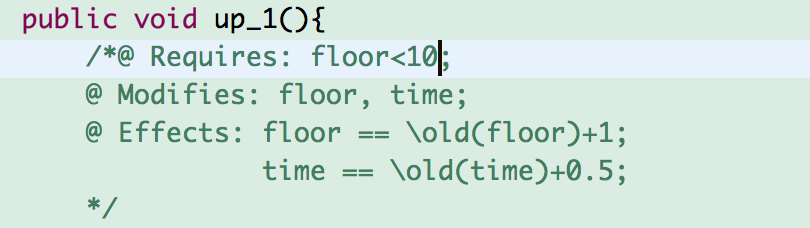
* + 1. Elevator()是构造函数，构造一个初始在1楼、时间为0的电梯。满足不变式。



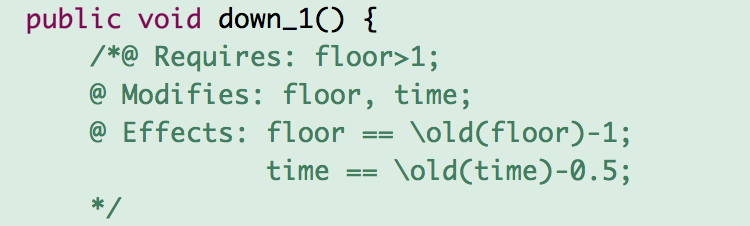
* + 1. getFloor、getTime为查询方法，不会修改对象，因此不会影响repOK.
    2. setTime(double t)方法是把time赋值为t，由于前置条件保证了t>=0，因此变更后仍然满足不变式。
    3. setDir只会修改dir，不会改变rep，仍然满足不变式。
    4. toString返回电梯的时间和楼层信息的字符串，不会改变rep，仍然满足不变式。
    5. Down\_1,Up\_1分别为向下一楼和向上一楼，在前置条件的保证下仍然使得floor满足不变式，时间加上0.5仍然满足>=0。Open函数将time加上1，满足不变式。

所以，所有方法执行前和执行后的repOK都为true。综上，对该类任意对象的任意调用都不会改变其repOK为true的特性。因此该类任意对象始终保持对象有效性。

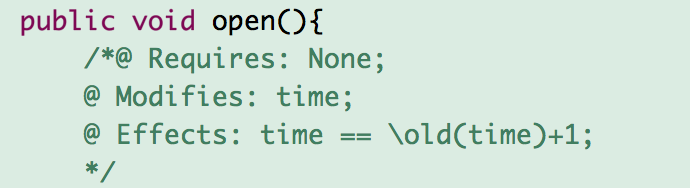
## 方法实现正确性



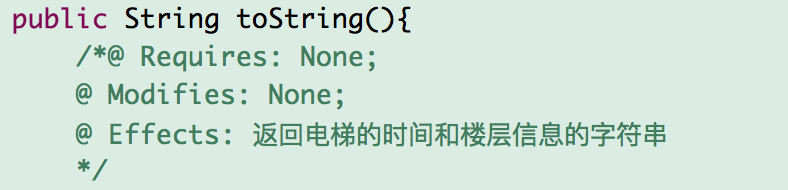
<this.floor++, this.time+=0.5> with <any>



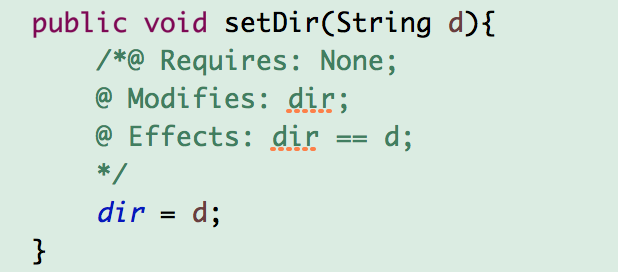
<this.floor--, this.time-=0.5> with <any>



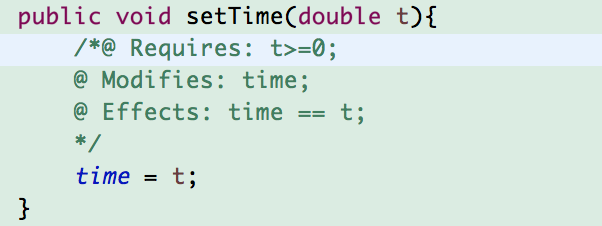
<this.time++> with <any>



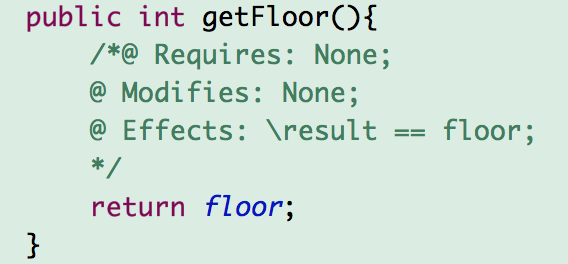
<return a string with information of floor and time> with <any>



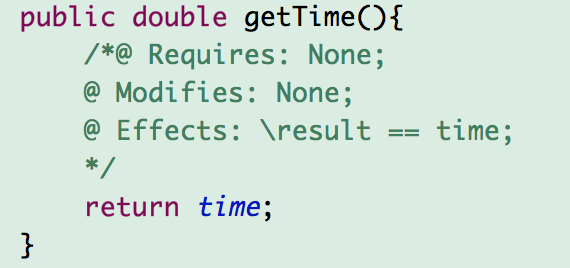
<set dir to d> with <any>



<set time to t> with <any>



<get floor> with <any>



<get time> with <any>

综上所述，所有方法的实现都满足规格。从而可以推断，elevator的实现是正确的，即满足其规格要求。

# 请求队列类（RequestQueue）

## 3.1 抽象对象有效实现



表示对象为存放着时间、楼层、上下、类型的列表，通过抽象函数映射为请求的各个信息的ArrayList。

## 3.2 对象有效性

* + 1. RequestQueue()是构造函数，新建4个ArrayList对象，代表请求队列存放的请求的信息。这样的话，rep得到满足，即满足了不变式。
    2. getArray\_time、getArray\_floor、getArray\_updown、getArray\_type为返回这几个ArrayList的数组形式，getCnt为返回这些ArrayList的大小，这些方法都不会修改对象，因此不会影响repOK.
    3. rq\_add为在这几个ArrayList里面增加上req中的信息，因此这几个ArrayList仍然非null，仍然满足rep.

所以，所有方法执行前和执行后的repOK都为true。综上，对该类任意对象的任意调用都不会改变其repOK为true的特性。因此该类任意对象始终保持对象有效性。

## 3.4方法实现正确性



<return Long[] of list\_t> with <any>



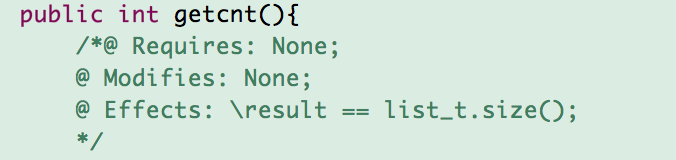
<return int[] of list\_f> with <any>



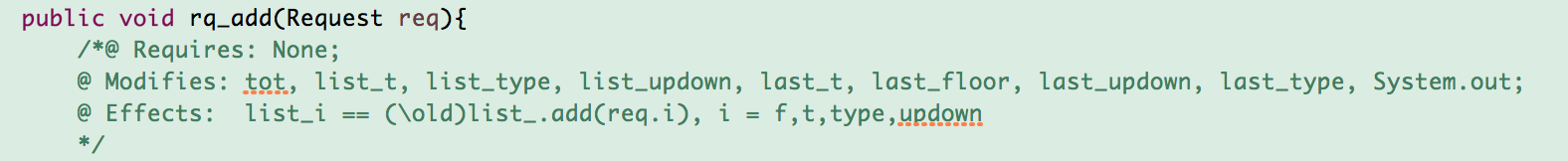
<return int[] of list\_updown> with <any>



<return int[] of list\_type> with <any>



<return list\_t.size()> with <any>



<list\_updown.contains(req.getupdown(i))> with <any>

<list\_f.contains(req.getfloor(i))> with <any>

<list\_type.contains(req.gettype(i))> with <any>

<list\_t.contains(req.gett(i))> with <any>

把几个ArrayList都添上请求i的各个信息。

综上所述，所有方法的实现都满足规格。从而可以推断，RequestQueue的实现是正确的，即满足其规格要求。