**ele项目（版本一）**

**任务**

(1) 设计实现电梯类、 乘客类以及你认为必要的类。

(2) 设计实现显示仿真信息的界面：显示每部电梯的状态（ 空闲或运行中）， 运行的方向（上行或者下行），目前停留的楼层；电梯内乘客的数量， 每位乘客要去往的楼层（ 例如12-32表示标识为12的乘客要求去往32层） ；当前时间。

(3) 假设10部电梯各自独立运行， 均可到达每一层。

(4) 随机产生每位乘客所要到达的楼层（ 假设每位乘客只产生一次乘梯请求） ， 并随机选择一部电梯让其等待。

(5) 实现满足该要求的电梯仿真流程， 选择电梯将乘客送至其请求的楼层。当所有乘客均到达请求的楼层后， 仿真结束。

该电梯系统由四个类组成，分别是Building, Floor, Elevator和Rider。

**1. main函数**

main函数主要用来开始模拟电梯系统（假设每秒有5个Rider需要乘坐电梯，并且只在前20秒产生Rider）

模拟停止条件：每一层都没有Rider进行等待，并且电梯内部没有Rider

**2. Building类**

成员变量：

time（int）：用来记录程序开始到现在所进行的时间

floor（vector<Floor>）：将Building中的所有Floor进行合并成一个集合

elevator（vector<Elevator>）：将Building中的所有Elevator进行合并成一个集合

成员函数：

构造函数：用于初始化elevator和floor容器

Building& step（int）：用于有Rider产生时进行的模拟

Building& step（）：用于无Rider产生时进行的模拟

void run（）：对elevator以及floor状态进行更改

友元函数：

过载 << 运算符，使其可以右端参数可以为Building类

设计思路：

每秒钟分别对floor和elevator进行遍历，并修改每个floor和elevator的状态。

如果floor中存在等待的Rider，那么就会遍历所有的elevator，在运行方向和Rider一致并且已到达或即将到达Rider所在楼层的所有elevator中更改它首先要到达的楼层为该floor。

如果elevator正在运行，它会先到达首先要到达的楼层，然后更新电梯内Rider以及新的首先要到达的楼层。

**3. Floor类**

成员变量：

upRiders（vector<Rider>）：用于储存上行Rider

downRiders（vector<Rider>）：用于储存下行Rider

location（int）：标志电梯所在位置

成员函数：

构造函数：初始化floor的楼层

bool isPreferredDirectionUp（）：判断floor是否需要呼叫方向向上的电梯

void addNewRider(const Rider&)：向floor添加Rider

vector<Rider> removeUpRiders(int)：将满足条件的upRiders从floor移除

vector<Rider> removeDownRiders(int)：将满足条件的downRiders从floor移除

**4. Elevator类**

成员变量：

IDLE/UP/Down（int）：电梯运行状态

capacity（int）：规定电梯最大容量

speed（int）：规定电梯运行速度

toFloor（Floor\*）：记录首先要到达的楼层

location（int）：记录当前位置

direction（int）：记录电梯运行方向

rider（vector<Rider>）；用于盛放elevator内Rider成员

成员函数：

构造函数：初始化elevator容量，速度，以及最初位置

vector<Rider> removeRidersForDestinationFloor()：将已经到达目标floor的Rider移除rider中

void addRiders(const vector<Rider>&)：向elevator中添加Rider成员

void setDestinationBasedOnRiders()：取elevator中Rider距离最近的floor设为目的地

bool isNearDestination() const：判断是否到达目标楼层

void moveToDestinationFloor()：将elevator位置设为目标floor位置

友元函数：

过载 << 运算符，使其可以右端参数可以为Elevator类

**5. Rider类**

成员变量：

copyID（static int）：标记Rider编号

f（const Floor\*）：记录Rider首先要到达的楼层

成员函数：

构造函数：初始化Rider所在楼层以及RiderID

Rider& operator=(const Rider&)：运算符=过载，因为其中成员变量存在地址变量

bool operator<(const Rider&) const；运算符<过载，用于区分Rider优先级

**ele项目（版本二）**

**任务**

修改乘客乘梯请求规则

随机产生每位乘客初次所要到达的楼层，并随机选择一部电梯让其等待；

每位乘客乘坐电梯达到指定楼层后，随机地停留10-120秒后，再随机地去往另一楼层，依此类推。当每位乘客乘坐过L次（每人的L值不同，在产生乘客时随机地在1-10之间确定）电梯后，第L+1次为下至底层并结束乘梯行为。到所有乘客结束乘梯行为时，本次仿真结束。

**修改内容**

版本一中Building::run()方法改名为Building::ElevatorRun()

main方法中程序结束标志改为Building::isRider[100]全部都为false

在Rider类中添加成员变量：

floor\_queue（std::queue<int>）：是一个用来排列Rider需要去向的floor队列

isinDestination（bool）：用来判断Rider时候到达指定楼层

waitime（int）：指定Rider在指定楼层等待时间

在Building类中添加成员变量：

isRider[100]（bool）用于判断Rider是否最终回到了1层（根本用途是作为程序结束标志）

在Building类中添加方法：

void getDifferentInts(int max, std::queue<int >&q)：产生随机数，并把它们放入Rider所要去往楼层的队列floor\_queue中

void run(vector<Rider> &,vector<Floor\*> &):[版本二核心内容]使用迭代器遍历每一个Rider，如果他的isinDestination为true，即到达目标楼层，那就让他等待waitime长度时间，并且更新他要到达下一目标floor信息

**ele项目（版本三）**

**任务**

修改电梯运行规则

E0、E1：可到达每层；

E2、E3：可到达1、25~40层。

E4、E5：可到达1~25层。

E6、E7：可到达1、2~40层中的偶数层。

E8、E9：可到达1~39层中的奇数层。

按照相同规则运行的两部电梯之间是联动的。

随机产生每位乘客初次所要到达的楼层，选择一部合适的电梯让其等待。

**修改内容**

在Floor和Elevator类中添加can\_move[10][40]（bool），用来表示第i个电梯在t层能否停靠（can\_move[i][t]），然后在Rider进入电梯时通过判断Rider的目的地在该电梯是否能够停靠。

具体实现：对vector<Rider> Floor::removeUpRiders(int max,int num)和vector<Rider> Floor::removeDownRiders(int max,int num)进行更改，增加判断Rider的目的地在该电梯是否能够停靠。

**ele项目（版本四）**

**任务**

电梯运行统计分析

仿真结束后，对电梯在仿真过程中的运行情况进行统计分析。

统计各梯的运行与空闲时间；统计各乘客发出乘梯要求后的等待时间；仿真结束后显示这些时间。

**修改内容**

在Rider类中添加成员变量：

wait\_time（int）：记录Rider在电梯外的等待时间

在Elevator类中添加成员变量：

run\_time（int）：记录Elevator的运行时间

在void Building::run(vector<Rider> &riders,vector<Floor\*>&floor)中增加对每一个Rider在电梯外时间的记录

在void Building::ElevatorRun() 中增加Elevator的运行时间

**ele项目（版本五）**

**任务**

第五六次上机内容：电梯高峰期预警和推荐功能（选做）

根据对电梯仿真运行的统计分析结果，推测电梯运行的高峰期，给在此阶段内有乘梯需求的乘客提前预警或推荐电梯较为空闲的时间段。

提示：在随机产生乘容所去的楼层时，加上时间因素，在不同时刻随机产生的乘容数量及目的地楼层体现出较大差别，这样比较容易做出预测。

高峰期预警和推荐功能，两个选择一个实现即可

**修改内容**

通过matplotlib.pyplot实现电梯内部Rider数据可视化

在main.cpp文件中新建函数

void plot(vector<int> vector1, const int n, vector<int> vector2)：用于画图的函数

string arr\_to\_string\_list(vector<int> vector1, const int n)：用于将vector<int>数据转化成string类型数据

void pythonInitial();//用于python初始化