UML Design

Class Passenger

private:

// Inherent info

string PID

int originFloor;

int destFloor;

bool served;

// Operation-related info

string in_or_out_car = "Out";

HallwayButton HBtn;

// Time related operation

Clock TimeRequested

Clock WaitTimer

Clock TravelTimer

public

Passenger()

Passenger(string, int, int, int)

// Set get functions

setPID(String);

setOriginFloor(int)

setDestFloor(int)

getPID(): String

getOriginFloor(): int

getDestFloor(): int

getServed(): bool

setServed(bool)

ChangeStatusofServed()

// Time related

void waitTimePasses()

void travelTimePasses()

void setWaitTime(int)

void setTravelTime(int)

Clock getWaitTime(): Clock

getTravelTime(): Clock

void ShowWaitTime()

void ShowTravelTime()

void SetTimeRequested(int, int, int)

getTimeRequested(): Clock

void set_in_or_out_car(string)

get_in_or_out_car(): String

setHallwayButton();

pushHallwayButton(): HallwayButton

GetIn()

GetOut()

Overloading == & != & <<

GenerateReport()

UML

~~Class~~ Struct Motor

AccelerateUp(): String
DecelerateUp(): String
AccelerateDown(): String
DecelerateDown(): String
MovingConstantSpeedDown(): String
MovingConstantSpeedUp(): String()
Stop(): String

~~Class~~ Struct HallwayButton

~~Pavu~~

bool UpButton;
bool DownButton;
HallwayButton()

Class Clock

public
Clock()
Clock(int)
Clock(int, int, int)
setTimer(int)
getTimer(): int
dispTime();
add(int)
subtract(int)
getSeconds(): int
getMinutes(): int
getHours(): int
setSeconds(int);
setMinutes(int)
setHours(int)

Private:
int minutes
int seconds
int hours

UML Design

Class ServiceQueue

private :

struct ListNode

T value
structListNode * next

ListNode * head;
ListNode * ptr

public :

ServiceQueue ()
appendNode (T)
showQueue ()
isIn (T) : bool
insertNodeUp (int)
insertNodeDown (int)
dequeue ()
peekhead () : int
IsEmpty (): bool
deleteNode (T~~int~~):
reset ()
size () : int
seeAt (int) : T
seeNext () : ListNode *
seeCurrent () : T
seeCurrentPointer () : ListNode *

UML

Class ElevatorCar

Private

// Increment info
String CarID;
int CurrentFloor
// operation related
ServiceQueue<int> ServiceQueueUp
ServiceQueue<int> ServiceQueueDown
ServiceQueue<int> ServiceQueueCurrent
String TripDirection;
int State;
// Time-related info
Clock Timer
Clock StateTimer
int TripCounter
int MaxWaitTime
int MaxTravelTime
double TotalWaitTime
double TotalTravelTime
HallwayButton CarHBtn
Motor CarMotor

Public

Service<Passenger> QueueofPassengerRequested;
vector<Passenger> ListPassengerServed;
ElevatorCar()
ElevatorCar(String, int)

Class ElevatorCar

Count()
SetCarID(String)
getCarID(): String
setCurrentFloor(int)
int getCurrentFloor()
SetTripDirection(String)
SwitchTripDirection()
getTripDirection(): String
setState(int)

// Action Based on each state of car:
WaitIdle(): String
Accelerate(): String
ConstantMotion(): String
Decelerate(): String
OpensDoor(): String
getState()
setTimer(int)
getTimer()
ShowServiceQueueCurrent()
ShowServiceQueueUp()
ShowServiceQueueDown()

// Functions will help elevator run
HallwayOrderProcessed(Passenger)
ControlOrderProcessed(Passenger)
CheckCurrentFloorforRequests(vector<Passenger

MoveFloor2Floor()
MotorInteraction()
timePassesPassengers(Vector<Passenger
// Operational Functions
BeginningtoMoveUporDown(vector<Passeng
ElevatorSmallTrip(vector<Passenger>
ElevatorBigTrip(vector<Passen
GeneratePassengerProfiles(vector<Passen
GenerateElevatorReport(vector<Passe

ServiceQueue<int> getServiceQueueUp()
ServiceQueue<int> getServiceQueueDown()
ServiceQueue<passenger> getServiceQueueCurrent()

CalculateWaitTime()
CalculateMaxTravelTime()