

Project 4: Elevator Simulator

Background:

Really Awesome Dudes incorporated, RAD, is a well-known established corporation in California. They produce consumer goods but specialize in Foo and Bar. In light of the high demand of Foo and Bar, RAD has hit it big. So big, they expanded their workforce as well as established a new headquarters. RAD has just recently cleared a proposal for installing an elevator within their new ten story office building. However most the people in the company have never seen or never used an elevator. President Duderino of RAD has hired you as a simulations engineer based on Professor McPherson's recommendation. Your task is to simulate this elevator in its everyday use.

Details:

For this project, you will be given a header file along with an “input.txt” file and an “output.txt”. The format of the input file will be straight forward. Each line will contain an elevator command and maximum of 2 parameters tab separated. Comment lines can be found in the input file, they will start with '#' and should be ignored.

Commands:

The commands go as follows:

Start

start<tab>passengers<tab>floor

This will be the first command, you should set the initial state of the elevator with passengers and floor given with this command.

Move

move<tab>direction

Tells the elevator to move in a certain direction. Direction will be either UP or DOWN. The elevator should not exceed the basement nor the 10th floor.

Add

add<tab>number

Adds a number of passengers to the number of existing passengers. Should not exceed the maximum capacity.

Remove

remove<tab>number

Removes a number of passengers from the number of existing passengers. Should not exceed the minimum capacity.

Show

show

Display the current status of the elevator which includes number of passengers and current floor.

Exit

exit

Stops simulation altogether, regardless if there is more in the input file.

Requirements

You are required to keep track of elevator at all times because the show command can be placed anywhere within the file. If the number of passengers or floor reaches maximum or minimum capacity, do display notice message, but carry on the simulator to the point reasonable. EX. If the number of passengers is seven and the max is ten and five people are added, only add three of the five. Same applies for the floors. The header file (below) has a four function definitions which you are required to implement. **When parsing the input you must use the getline function and parse each line using string manipulation.**

Header File

```
/*
 * Programmer:  Tony McNevin (amcnevin)
 * Program:     Elevator Simulator
 * Date:        07/14/06
 * Purpose:     To simulate an elevator moving up/down
 *              and pick up / drop off passengers
 */

#include <iostream>
#include <cstdlib>
#include <string>
#include <fstream>

using namespace std;

const int FLOOR_MAX = 10; // maximum floor
const int FLOOR_MIN = 0; // basement is min
const int PASS_MAX = 10; // max of 10 passengers
const int PASS_MIN = 0; // cannot have negative passengers

void printStatus(ostream& out, const int floor, const int passengers); // Print elevator status
void printMsg(ostream& out, string msg); // Display message
void setPassengers(int pass, int& passengers); // Passenger mutator
void setFloor(string dir, int& floor); //Floor mutator
void setFloor(int init, int& floor); // Floor mutator
```

Tips

Because this application requires file parsing for commands and parameters, string manipulation will be stressed. The following string functions may help you: *getline*, *substr*, *find*, and *atoi*. If you want to get really technical look into *string streams*.

Submission

When you have completed this project and tested it thoroughly, you will submit you CPP file to the Curator and sign up for a time for a demo. There will be a sign up sheet outside of the McBryde 116/118 Lab. The demo will consist of sitting down with the TA and running your program with a predetermined set of input files. Grading will consist of accuracy of solution but also software engineering aspects.

Honor Code

Place this in the bottom of your file in comments

```
//- I have not discussed the C++ language code in my program with
// anyone other than my instructor or the teaching assistants
// assigned to this course.
//
// - I have not used C++ language code obtained from another student,
// or any other unauthorized source, either modified or unmodified.
//
// - If any C++ language code or documentation used in my program
// was obtained from another source, such as a text book or course
// notes, that has been clearly noted with a proper citation in
// the comments of my program.
//
// - I have not designed this program in such a way as to defeat or
// interfere with the normal operation of the Curator System.
//
// <Student Name>
```