```cpp
/*  Program: A3P4 - Elevator Sim
    Author: Tom Stutler
    Last Date Modified: 3/19/15

    The intent of this program is to display and track the position of three elevators, starting at floor 1,
    to a user that then picks one of the three elevators to operate, telling it which floor, 1-10, it wants to go to.

    This program assumes that the user is always on floor 1 regardless of the elevator the user picked to operate.
*/
//#define NDEBUG
#include <iostream>
#include <cassert>
using namespace std;

const int NUM_OF_ELEVATORS = 3;
class elevator
{
public:
    elevator() : position(1), destination(0) {}
    ///Initializes position to 1.
    void serviceRequest(int destinationParam);
    ///Moves the elevator from position to destination.
    int returnPosition();
    ///Returns the current position of the elevator.

private:
    int position;
    int destination;

    void showStatus();
    ///Displays the current action of the elevator.
};

void displayElevators(elevator elevatorsParam[]);
///Displays the current status of the three elevators.

int main()
{
    elevator elevators[NUM_OF_ELEVATORS];
    int elevatorChoice, floorChoice;
    bool repeat;

    do
    {
        //Show the user the current state of the elevators.
        displayElevators(elevators);

        //Prompt user to pick an elevator or exit.
        cout << "Which elevator do you want (1=A, 2=B, 3=C, or other to exit)? ";
        cin >> elevatorChoice;

        //Check if elevator is at floor 1, if not move it to floor 1.
        if ((elevators[elevatorChoice-1].returnPosition() > 1) && (elevators[elevatorChoice-1].returnPosition() <= 10)) {
            assert(elevators[elevatorChoice-1].returnPosition() != 1);
            elevators[elevatorChoice-1].serviceRequest(1);
        }

        if ((elevatorChoice == 1) || (elevatorChoice == 2) || (elevatorChoice==3)) {
            //Prompt user for a floor to go to.
            cout << "Which floor do you want? ";
            cin >> floorChoice;

```

```
                //Move the human to its petty destination.
                elevators[elevatorChoice-1].serviceRequest(floorChoice);
                repeat = true;
        } else {
            repeat = false;
        }
    } while (repeat == true);
}

void displayElevators(elevator elevatorsParam[])
{
    cout << "Elevator Status\n"
         << "A\tB\tC\n";

    for (int i=0; i < NUM_OF_ELEVATORS; i++) {
        cout << elevatorsParam[i].returnPosition()
             << "\t";
    }

    cout << "\n";
}

void elevator::serviceRequest(int destinationParam)
{
    destination = destinationParam;

    if (position < destination) {
        cout << "Starting at floor "
             << position
             << endl;

        for (;position < destination; position++) {
            showStatus();
        }
    } else if (position > destination) {
        cout << "Starting at floor "
             << position
             << endl;

        for (;position > destination; position--) {
            showStatus();
        }
    } else {
        cout << "The elevator is already on that floor.\n";
    }
    showStatus();
}

int elevator::returnPosition()
{
    return position;
}

void elevator::showStatus()
{
    if (position < destination) {
        cout << "Going up - now at floor "
             << position+1
             << endl;
    } else if (position > destination) {
        cout << "Going down - now at floor "
             << position-1
             << endl;
    } else {
        cout << "Stopping at floor "
             << position
             << endl;
    }
}
```