UNBC CPSC 101 Team Term Project Winter 2011

This document contains a problem description statement for the 2011 Winter CPSC 101 course. The problem statement has been taken almost verbatim from a previous textbook used when the course was taught in C++.

Instructions on what to do with this problem statement are (or will be) provided separately.

This version of the problem statement was printed January 14, 2011.

1 Problem Statement

A company intends to build a two-floor office building and equip it with an elevator. The company wants you to develop an object-oriented *software simulator* in Java that models the operation of the elevator to determine whether the elevator suits the company's needs.

Your simulator should include a clock that begins with its time, in seconds, set to zero. The clock ticks (increments the time by one) every second, but it does not keep track of the hours and minutes. Your simulator also should include a scheduler that begins the day by scheduling two times randomly: the time when a person first steps onto floor 1 and the button on that floor to summon the elevator, and the time when a person first steps onto floor 2 and presses the button on that floor to summon the elevator. Each of these times is a random integer in the range from 5 to 20 seconds, inclusive (i.e., 5, 6,7, . . . , 20). When the clock time equals the earlier of these two times, the scheduler creates a person, who then walks onto the appropriate floor and presses the floor button. [*Note*: It is possible that these two randomly scheduled times will be identical, in which case people step onto both floors and press both floor buttons at the same time.] The floor button illuminates, indicating that it has been pressed. [*Note*: The illumination of the floor button occurs automatically when the button is pressed and needs no programming; the light built into the button turns off automatically when the button is reset.] At the beginning of the simulation, the elevator starts the day waiting with its door closed on floor 1. To conserve energy, the elevators moves only when necessary. The elevator alternates directions between moving up and moving down.

For simplicity, the elevator and each of the floors have a capacity of one person. The scheduler first verifies that a floor is unoccupied before creating a person to walk onto that floor. If the floor is occupied, the scheduler delays creating the person by one second (thus allowing the elevator an opportunity to pick up the person and move to the floor). After a person walks onto a floor, the scheduler creates the next random (between 5 and 20 seconds into the future) for a person to walk onto that floor and press the floor button.

When the elevator arrives at a floor, it resets the elevator button and sounds the elevator bell (which is inside the elevator). The elevator then signals its arrival to the floor. The floor, in response, resets the floor button and turns on the floor's elevator-arrival light. The elevator then opens its door. [*Note*: The door on the floor opens automatically with the elevator door and needs no programming.] The elevator's passenger, if there is one, exits the elevator, and a person, if there is one waiting on that floor, enters the elevator. Although each floor floor has a capacity of one person, assume there is enough room on each floor for a person to wait on that floor while the elevator's passenger exits.

A person entering the elevator presses the elevator button, which illuminates (automatically, without programming) when pressed and turns off when the elevator arrives on the floor and resets the elevator button. [*Note*: Because the building has only two floors, only one elevator button is necessary; this button notifies the elevator to move to the other floor.] Next, the elevator closes its door and begins moving to the other floor. When the elevator arrives at a floor, if a person does not enter the elevator and the floor button on the other floor has not been pressed, the elevator closes its door and remains on that floor until another person presses a button on a floor.

For simplicity, assume that all the activities that happen, from when the elevator reaches a floor until the elevator closes its door, take zero time. [*Note*: Although these activities take zero time, they still occur sequentially; e.g., the elevator door must open before the passenger exits the elevator.] The elevator takes five seconds to move from one floor to the other. Once per second, the simulator provides the time to the scheduler and to the elevator. The scheduler and elevator use the time to determine what actions each must take at that particular time, [sic] (e.g., the scheduler might determine that it is time to create a person, and the elevator, if moving, might determine that it is time to arrive at its destination floor).

The simulator should display messages on the screen that describe the activities that occur in the system. These include a person pressing a floor button, the elevator arriving on a floor, the clock ticking, a person entering the elevator, etc.