Grand Hotel

Grand Hotel has a number of areas (at most 20) organized in 50 floors (1, .. 50), with the front reception at floor 1 (ground floor).
Each floor has a unique 4-symbol alphanumeric identification , and contains a number of rooms (at most 30).
Each room has a numeric identification, and can accommodate a limited number of persons (there are a number of beds in the room, between 1 and 6).
Persons are identified by their Social Security number (SSN, 13 digit unique number), name, and birth date (4-digit year, 2-digit month, 2 digit day). There are persons which are out for the moment, and thus they are not currently present in any room. When a person returns to the hotel, he/she should be allowed in the same room where he/she was hosted before getting out.

Two static methods is provided, in the package **`hotel,`** class **`Provided`**:

`generatePersonInfo:` which generates information about a new person, and returns an array of strings with SSN (13 digit unique number), name, and birth date (4-digit year, 2-digit month, 2-digit day) in this order, and has no arguments.

generateRoomInfo: which generates info for a hotel room, and returns and array of strings with identification and number of beds in this order. No arguments as well.

Develop a stand-alone application which supports the following operations:

1. Populate the hotel with guests
2. Make a number of guests get out for a walk (at random)
3. Make a specific person (identified by SSN) leave the hotel
4. List all the persons born on the same date, complete with location in the hotel
5. List all the persons hosted at a given floor, in a given room
6. List all the persons who went for a walk
7. Place a person in the first available room (remember there are persons who are temporary out), as close as possible to other persons who were born on the same date
8. Find an guest hosted in the hotel indicating the floor and room (note that he/she might be out momentarily)