# **HOSTEL MANAGEMENT SYSTEM**

import java.util.HashMap;

import java.util.Map;

class Hostel {

private String name;

private int totalBuildings;

private int floorsPerBuilding;

private int roomsPerFloor;

private String[] requirementsPerRoom;

private Map<String, Map<String, Object>> students;

private Map<String, Map<String, Object>> rooms;

public Hostel(String name, int totalBuildings, int floorsPerBuilding, int roomsPerFloor, String[] requirementsPerRoom) {

this.name = name;

this.totalBuildings = totalBuildings;

this.floorsPerBuilding = floorsPerBuilding;

this.roomsPerFloor = roomsPerFloor;

this.requirementsPerRoom = requirementsPerRoom;

this.students = new HashMap<>();

this.rooms = initializeRooms();

}

private Map<String, Map<String, Object>> initializeRooms() {

Map<String, Map<String, Object>> rooms = new HashMap<>();

for (int building = 1; building <= totalBuildings; building++) {

for (int floor = 1; floor <= floorsPerBuilding; floor++) {

for (int room = 1; room <= roomsPerFloor; room++) {

String roomId = building + "-" + floor + "-" + room;

Map<String, Object> roomInfo = new HashMap<>();

roomInfo.put("occupied", false);

roomInfo.put("requirements", requirementsPerRoom.clone());

roomInfo.put("occupant", null);

rooms.put(roomId, roomInfo);

}

}

}

return rooms;

}

public String allocateRoom(String studentId, String[] requirements) {

for (Map.Entry<String, Map<String, Object>> entry : rooms.entrySet()) {

String roomId = entry.getKey();

Map<String, Object> roomInfo = entry.getValue();

if (!(Boolean) roomInfo.get("occupied") && containsAllRequirements(roomInfo, requirements)) {

roomInfo.put("occupied", true);

roomInfo.put("occupant", studentId);

Map<String, Object> studentInfo = new HashMap<>();

studentInfo.put("room\_id", roomId);

studentInfo.put("requirements", requirements.clone());

students.put(studentId, studentInfo);

return roomId;

}

}

return null;

}

public String reallocateRoom(String studentId, String[] requirements) {

Map<String, Object> studentInfo = students.get(studentId);

if (studentInfo != null) {

String currentRoomId = (String) studentInfo.get("room\_id");

rooms.get(currentRoomId).put("occupied", false);

rooms.get(currentRoomId).put("occupant", null);

String newRoomId = allocateRoom(studentId, requirements);

if (newRoomId != null) {

studentInfo.put("room\_id", newRoomId);

return newRoomId;

}

}

return null;

}

private boolean containsAllRequirements(Map<String, Object> roomInfo, String[] requirements) {

for (String req : requirements) {

if (!containsRequirement(roomInfo, req)) {

return false;

}

}

return true;

}

private boolean containsRequirement(Map<String, Object> roomInfo, String requirement) {

return ((String[]) roomInfo.get("requirements")).contains(requirement);

}

public int getEmptyRoomsCount() {

int emptyRooms = 0;

for (Map<String, Object> roomInfo : rooms.values()) {

if (!(Boolean) roomInfo.get("occupied")) {

emptyRooms++;

}

}

return emptyRooms;

}

public int getStudentsCountByInstitute(String institute) {

int studentsCount = 0;

for (Map<String, Object> studentInfo : students.values()) {

if (containsRequirement(studentInfo, institute)) {

studentsCount++;

}

}

return studentsCount;

}

public void generateReport() {

System.out.println("Hostel Name: " + name);

System.out.println("Total Buildings: " + totalBuildings);

System.out.println("Floors per Building: " + floorsPerBuilding);

System.out.println("Rooms per Floor: " + roomsPerFloor);

System.out.println("Requirements per Room: " + Arrays.toString(requirementsPerRoom));

System.out.println("Total Rooms: " + rooms.size());

System.out.println("Occupied Rooms: " + students.size());

System.out.println("Empty Rooms: " + getEmptyRoomsCount());

}

}

public class HostelManagementSystem {

public static void main(String[] args) {

Hostel hostelA = new Hostel("Hostel A", 2, 3, 5, new String[]{"Male", "Engineering"});

hostelA.allocateRoom("S001", new String[]{"Male", "Engineering"});

hostelA.allocateRoom("S002", new String[]{"Female", "Arts"});

hostelA.generateReport();

hostelA.reallocateRoom("S001", new String[]{"Female", "Engineering"});

hostelA.generateReport();

}

}