

```

1. public class Room {
2.
3.     private String code;
4.     private int capacity;
5.     private static Room[] rooms = new Room[100];
6.     private static int numRooms = 0;
7.     /**
8.      * The constructor for the Room class.
9.      * This is a class used to represent the rooms used for
      COVID tests.
10.     *
11.     * @param code this is the code of the room to
      identify it.
12.     * @param capacity this is the number of assistants
      that can be allocated to a
13.     room.
14.     */
15.     public Room(String code, int capacity){
16.         this.code = checkCode(code);
17.         this.capacity = checkCapacity(capacity);
18.         addRooms(this);
19.         iterateNumRooms();
20.     }
21.     /**
22.     * The getter method for the room code
23.     *
24.     * @return code this is the code of the room to
      identify it.
25.     */
26.     public String getCode(){
27.         return code;
28.     }
29.     /**
30.     * The getter method for the capacity of the room.
31.     *
32.     * @return capacity this is the number of assistants
      that can be allocated to a
33.     room.
34.     */
35.     public int getCapacity(){
36.         return capacity;
37.     }
38.     /**
39.     * The getter method for the array of all rooms.
40.     *
41.     * @return the array of all rooms.
42.     */
43.     public static Room[] getRooms(){
44.         return rooms;
45.     }
46.     /**
47.     * The getter method for then number of rooms.
48.     *
49.     * @return the number of rooms.

```

```

50.      */
51.      public static int getnumRooms(){
52.          return numRooms;
53.      }
54.      /**
55.       * The setter method for the code of the room.
56.       *
57.       * @param code this is the code of the room to
        identify it.
58.       */
59.       public void setCode(String code){
60.           this.code = checkCode(code);
61.       }
62.       /**
63.       * This is the setter method for the capacity of the
        room.
64.       *
65.       * @param capacity this is the number of assistants
        that can be allocated to a
66.       room.
67.       */
68.       public void setCapacity(int capacity){
69.           this.capacity = checkCapacity(capacity);
70.       }
71.       /**
72.       * This method increases the number of rooms by one
        when called.
73.       */
74.       private static void iterateNumRooms(){
75.           numRooms += 1;
76.       }
77.       /**
78.       * This method is used to add new rooms to the array
        of rooms.
79.       *
80.       * @param room this is a instance of a room.
81.       */
82.       private static void addRooms(Room room){
83.           rooms[numRooms] = room;
84.       }
85.       /**
86.       * This method will check if the code is unique.
87.       *
88.       * @param code this is the code being tested.
89.       * @return this code is returned only if the code is
        unique.
90.       */
91.       private String checkCode(String code){
92.           for (int i=0;i<numRooms;i++){
93.               if (rooms[i].getCode().equals(code)){
94.                   throw new IllegalArgumentException("Code is not
        unique.");
95.               }
96.           }

```

```

97.         return code;
98.     }
99.     /**
100.      * This method will check the capacity is greater than
        zero.
101.      *
102.      * @param capacity this is the number of assistants
        that can be allocated to a
103.      room.
104.      * @return capacity this is the number of assistants
        that can be allocated to a
105.      room.
106.      */
107.     private int checkCapacity(int capacity){
108.         if (capacity < 0){
109.             throw new IllegalArgumentException("Capacity should
        be greater than
110.         zero.");
111.         }
112.         return capacity;
113.     }
114.     /**
115.      * This method will return a string of all the room's
        parameters.
116.      *
117.      * @return a string of all the room's parameters
118.      */
119.     public static String toStringAll(){
120.         String allRooms = "Rooms-\n";
121.         for (int i=0;i<numRooms;i++){
122.             allRooms = allRooms.concat((i+11)+".
        "+rooms[i].toString()+"\n");
123.         }
124.         return allRooms;
125.     }
126.     /**
127.      * This method will return the string of a room's
        parameters.
128.      *
129.      * @return a string of a room's parameters.
130.      */
131.     public String toString(){
132.         return "| "+code+" | capacity: "+capacity+" |";
133.     }
134. }

```