Questions 1 and 2 are based on the following scenario:

An object-oriented application, written in Java, is being developed for simulating a heating system for a small building in Canada.

The application has two major classes: **`Radiator`** and **`Room`**.

The following properties apply to the classes:

- A **`Radiator`** heats one **`Room`**.
- A **`Room`** can be heated by zero or **up to 2 unique** **`Radiators`**
- The **`Radiator`** class
  - has the state: **`radiatorID`** (int) and **`isTurnedOn`** (boolean).
  - has a constructor that does not accept any arguments.
  - generates unique integer values for the **`radiatorID`** starting from 1000 and increases in increments of 10 for each successive **`Radiator`** object.
  - has its state as **turned off** initially.
  - has one accessor method for the **`radiatorID`**
  - has one public method, **`heats(Room rm)`**, that associates a **`Room`** object with a **`Radiator`** object. No return values.
- The **`Room`** class
  - has the state: **`roomName`** (String), **`seatingCapacity`** (int) and **`numRadiators`** (int).
  - has a constructor that accepts a **`roomName`** and initialises the **`seatingCapacity`** to a default value of 15.
  - has one public method **`isHeatedBy(..)`** that accepts and associates a **`Radiator`** object with the **`Room`** object. This method returns a String containing one of the following messages indicating whether the **`Radiator`** was successfully associated with the **`Room`** or not:
    - "Radiator already added to room."
    - "Radiator successfully added to room."
    - "Cannot add Radiator. Room has maximum number of radiators."

Question 1 (10 marks): Classes, State, Behaviour

Write Java code for the **`Radiator`** class. Ensure that information hiding principles are enforced. Include additional variables as necessary for implementing the functionality described for the class. [10 marks]

Question 2 (15 marks): Relationships: Association, Aggregation and Composition

(a) Write Java code for the **`isHeatedBy(Radiator rd)`** method in the **`Room`** class based on the specifications described for the classes. Assume that the **`Room`** class has an array called **`radiators`** declared of type **`Radiator[]`**, and it can store two **`Radiator`** objects. Your solution should perform necessary error checking and return appropriate String messages. [8 marks]

(b) Write Java code to:
- Create one **`Room`** object, **`rm1`**, with arbitrary state, and two **`Radiator`** objects: **`rd1`**, **`rd2`**.
- Associate the objects such that the **`Room`** object **`rm1`** is heated by the two **`Radiator`** objects, and such that the **`Radiator`** objects are aware of the **`Room`** that they heat. [4 marks]

(c) Argue whether association, aggregation or composition relationships exist between the Room and Radiator classes with supporting evidence using code snippets from your answers. Identify the total number of occurrences of each relationship (if any). [3 marks]