**Question 1**

How do you indicate that a class represents an entity to store in a Room database?

* Make the class extend DatabaseEntity.
* Annotate the class with @Entity.
* Annotate the class with @Database.
* Make the class extend RoomEntity and also annotate the class with @Room.

**Question 2**

The DAO (data access object) is an interface that Room uses to map Kotlin functions to database queries.

How do you indicate that an interface represents a DAO for a Room database?

* Make the interface extend RoomDAO.
* Make the interface extend EntityDao, then implement the DaoConnection() method.
* Annotate the interface with @Dao.
* Annotate the interface with @RoomConnection.

**Question 3**

Which of the following statements are true about the Room database? Choose all that apply.

* You can define tables for a Room database as annotated data classes.
* If you return LiveData from a query, Room will keep the LiveData updated for you if the LiveData changes.
* Each Room database must have one, and only one, DAO.
* To identify a class as a Room database, make it a subclass of RoomDatabase and annotate it with @Database.

**Question 4**

Which of the following annotations can you use in your @Dao interface? Choose all that apply.

* @Get
* @Update
* @Insert
* @Query

**Question 5**

How can you verify that your database is working? Select all that apply.

* Write instrumented tests.
* Continue writing and running the app until it displays the data.
* Replace the calls to the methods in the DAO interface by calls to equivalent methods in the Entity class.
* Run the verifyDatabase() function provided by the Room library.