**Given Information:**

* **Class Horse** is a base class.
* **Class Equestrian** is a subclass of Horse (inherits from Horse).
* **Class Racer** is another subclass of Horse.

This means we have the following inheritance structure:

Horse

/ \

Equestrian Racer

**Key Points on Type Casting:**

1. **Upcasting (Safe Casting):**
   * A subclass object can always be treated as an instance of its superclass.
   * Example: If Equestrian inherits from Horse, then an Equestrian object can be treated as a Horse.
2. **Downcasting (Explicit Casting, Potentially Unsafe):**
   * Casting a superclass reference to a subclass requires explicit casting.
   * This only works if the object being cast actually belongs to the subclass type.

**The Scenario:**

* d is declared as type Horse, but it refers to an object of type Equestrian.
* We attempt:

Racer r = (Racer) d;

* Since d actually refers to an Equestrian object, and Equestrian and Racer are siblings (not in the same direct hierarchy), **this cast is invalid** and will cause a ClassCastException at runtime.

**Why Does It Fail?**

* **Equestrian and Racer are unrelated except through Horse**  
  Since Equestrian is **not** a subclass of Racer, and vice versa, casting between them is not allowed.
* **Java allows casting only when the actual object is an instance of the target class or its subclass.**  
  In this case, the object is an instance of Equestrian, not Racer.

**Conclusion:**

❌ **The cast (Racer) d; is invalid and will result in a ClassCastException at runtime.**  
✅ **Valid casting would be:**

* (Horse) d; (since Equestrian is a subclass of Horse)
* (Equestrian) d; (since d actually refers to an Equestrian)