Erklärung der Methodenaufrufe:

1. `System.out.println(u1.getRole());`
   - Ruft getRole() auf dem User-Objekt auf
   - Gibt “General user” zurück
2. `System.out.println(l1.getRole());`
   - Ruft getRole() auf dem Librarian-Objekt auf
   - Da die Methode in Librarian überschrieben wurde, wird die Librarian-Version verwendet
   - Gibt “UnknownLibrarian” zurück (name + “Librarian”)
3. `System.out.println(u2.getRole());`
   - u2 ist eine User-Referenz, die auf ein Librarian-Objekt zeigt
   - Aufgrund von dynamischer Bindung wird die überschriebene Methode in Librarian aufgerufen
   - Gibt “UnknownLibrarian” zurück
4. `l1.getName();`
   - Ruft getName() auf dem Librarian-Objekt auf
   - Da die Methode von User geerbt und nicht überschrieben wurde, wird die User-Version verwendet
   - Gibt “Unknown” zurück
5. `u1.getName();`
   - Ruft getName() auf dem User-Objekt auf
   - Gibt “Unknown” zurück
6. `l1.work();`
   - Ruft work() auf dem Librarian-Objekt auf
   - Führt die Methode aus (tut “some work”)
7. `u2.work();`
   - Dies würde einen Kompilierungsfehler verursachen
   - Obwohl u2 zur Laufzeit auf ein Librarian-Objekt zeigt, ist der statische Typ User
   - Die User-Klasse hat keine work()-Methode, daher ist der Aufruf nicht zulässig

   // Fehler: Falsche Erklärung des Kompilierungsfehlers
   - Der Fehler tritt auf, weil die work()-Methode nicht public ist

Warum einige Aufrufe nicht kompiliert werden können:

Der Aufruf `u2.work()` kann nicht kompiliert werden, weil: - Der statische Typ von u2 ist User - Die User-Klasse hat keine work()-Methode - Der Compiler prüft nur den statischen Typ, nicht den dynamischen Typ zur Laufzeit - Obwohl u2 zur Laufzeit auf ein Librarian-Objekt zeigt, das eine work()-Methode hat, erlaubt der Compiler den Aufruf nicht, da er nur den statischen Typ berücksichtigt

// Fehler: Falsche Behauptung über Typumwandlung - Man könnte das Problem lösen, indem man schreibt: ((User)u2).work()