Initialization Guarantees

Wir haben gesehen, dass am Ende der Initialisierung von Objekten nicht nur finale Felder in allen Threads sichtbar werden, sondern auch die transitive Hülle, der von diesen Referenzen erreichbaren Objekte.

Gegeben sind die Klassen Student und Address:

```
class Address {
   private String street;
   private String city;

   public Address(String street, String city) {
      this.street = street;
      this.city = city;
   }
   public String getStreet() { return street; }
   public String getCity() { return city; }
}

public class Student {
   private final String name;
   private Address address;

   public Student(String name, Address address) {
      this.name = name;
      this.address = address;
   }

   public String toString() {
      return String.format("%s %s %s", name, address.getStreet(), address.getCity());
   }
}
```

Aufgaben:

a) Ein Student wird wie folgt initialisiert:
`Student s = new Student("Müller", new Address("Bachweg 2", "Windisch"))`

Was sind die möglichen Resultate, wenn aus einem beliebigen Thread die Anweisung

```
if(s != null) System.out.println(s);
```

ausgeführt wird? Welche Werte sind garantiert sichtbar und welche nicht?

b) Beurteilen Sie die Situation wenn in der Klasse Student das Feld address ebenfalls final deklariert ist (nicht aber die Felder street und city in der Klasse Address):

c) Welche Resultate sind möglich, wenn Student wie folgt definiert ist:

```
public class Student {
   private String name;
   private Address address;
   private final Student me;

   public Student(String name, Address address) {
      this.me = this;
      this.name = name; this.address = address;
   }

   public String toString() {
      return String.format("%s %s %s", me.name,
                                        me.address.getStreet(), me.address.getCity());
   }
}
```