Hotel

Das File `hotels.db` enthält Informationen über Hotels[1]. Die Datei besitzt einen einzigartigen Aufbau. Es hilft, sich vorzustellen, mit einem binär gespeicherten Table aus einer Datenbank zu arbeiten; die Column-Namen sind die Properties eines Hotels.

- Die Datei beginnt mit einer ID(4 Byte),
- und danach einem Offset(4 Byte) `offset`.

1 Properties

Nach dem Offset folgt ein Block, welcher die Namen der Columns und die Column-Breiten definiert. Er beginnt mit der Anzahl an Columns(2 Byte). Danach folgen für **jede** Column folgende Informationen:

- Die Länge des Column-Namens in Byte(2 Byte) `n`
- Der Column-Name(`n` Byte)
- Die Breite der Column(2 Byte)

Lies diese Informationen aus und stelle sicher, dass der entsprechende Test passed.

2 Hotels

Schreibe nun eine Klasse `Hotel`, welche Instanzvariablen ähnlich den gelesenen Properties beinhaltet. Das Byte `offset` ist das erste Byte, welches Hotelinformationen beinhaltet.

- Vor jedem Hotel stehen 2 Byte, welche entscheiden, ob das Hotel als gelöscht markiert wurde oder nicht. Ist der gelesene Wert `0x0000`, so ist das Hotel gültig, `0x8000` bedeutet gelöscht. Andere Werte sind mit einer `IllegalArgumentException` abzuhandeln.
- Danach kommen die eigentlichen Hotelinformationen; die Breite der Columns wurde bereits ausgelesen und muss hier verwendet werden. Die Daten wurden allesamt zuerst in Strings verwandelt und dann binär geschrieben, also in etwa durch Code wie `output.write(property.toString().getBytes());`

3 Methoden

- `public static Map<String, Short> readProperties(String filename)`
  Liest die Properties(Column-Namen) aus und weist jeder die Bytelänge(Column-Breite) zu. Die **Reihenfolge** der Properties muss dabei so bleiben, wie in der Datei spezifiziert, da sonst das Auslesen der Hotels unmöglich wird.
- `public static Set<Hotel> readHotels(String filename) throws IOException`
  Liefert die Hotels aus der Datei sortiert nach Ort und dann nach Name.

[1] Quelle: Java Programmer Certification