Auswerten einer IBAN mit String-Funktionen

Hast du dich auch schon über den langen IBAN (International Bank Account Number) Code geärgert, den sich kein Mensch merken kann, welchen du aber auf deinen Banküberweisungen immer eintragen musst?

Aber weißt du auch, welche Informationen in der IBAN Nummer versteckt sind?(Quelle: Wikipedia):
DE68210501700012345678

- An Stelle 1 bis 2 steht das Länderkürzel
- An Stelle 3 bis 4 steht die Prüfziffer
- An Stelle 5 bis 12 steht die Bankleitzahl
- An Stelle 13 bis 22 steht die Kontonummer

Die IBAN bietet uns also ideale Möglichkeiten, um uns bei den Java Strings auszutoben.

Du sollst im Folgenden eine Klasse IBAN entwickeln.

- Wie übergeben zu Beginn die IBAN an die Klasse. Schreibe eine Methode die uns die Kontonummer, die Bankleitzahl, das Länderkürzel und die Prüfziffer zurückliefert.
- Eine weitere Methode soll die Informationen ausgeben. Hier die Programmausgabe:
  Land: DE
  Prüfziffer: 68
  Bankleitzahl: 21050170
  Kontonummer: 0012345678
- Schreibe dazu eine Methode, die testet, ob eine eingegebene IBAN die Bankleitzahl der Erzgebirgssparkasse enthält. Hierzu müssen wir überprüfen, ob eine IBAN den Teilstring 87054000 hat.
- Wir haben die Prüfziffer als String extrahiert. Aber sollte das nicht eigentlich ein Integer sein? Sehr gut aufgepasst! Wandle die Prüfziffer in eine Zahl um und speichere sie in einer Int-Variablen.
- Berechne in einer Methode nach der untenstehenden Anleitung die Prüfziffer der IBAN und gib aus, ob sie übereinstimmt.

1. Zunächst versetzen wir den Ländercode und die unbekannte Prüfziffer ans Ende der IBAN. Die Prüfziffer ersetzen wir durch zwei Nullen: „350700240388249600DE00“
2. Für jeden Buchstaben des Ländercodes gibt es eine zugeordnete Nummer. Der Buchstabe A bekommt die Nummer 10, der Buchstabe B die Nummer 11 und so weiter.
   Für DE ergibt sich: 13 und 14. Unsere IBAN sieht nun so aus: „350700240388249600131400“
3. Diese 24-stellige Zahl wird nun mit Modulo 97 der Rest ermittelt. In unserem Beispiel ist der Rest 54.
4. Dann subtrahieren wir den Rest von der ISO-genormten Zahl 98 → Prüfziffer. In unserem Beispiel ist dies 44.

Das Problem ist, dass eine 24 stellige Zahl nicht als int-Wert gespeichert werden kann.
Für solche riesigen Werte wurde in der Math-Bibliothek von Java die Klasse **BigInteger** entwickelt.
Schreibe über die Klasse: `import java.Math.*;`

```
BigInteger iban_BG = new BigInteger(iban_umgestellt); erstellt eine neues BigInteger-Objekt.

Die Funktionen add(BigInteger wert) subtract(BigInteger wert) multiply(BigInteger wert)
divide(BigInteger wert) remainder(BigInteger wert) pow(int exponent) berechnen die Summe,
Differenz, Produkt, Quotient, Modulo und Potenzvon BigInteger-Zahlen.

Bsp.: BigInteger divisor= new BigInteger("97");
      BigInteger rest = iban_umgestellt.remainer(divisor);

 xyz.intValue() wandelt den BigInteger Wert xyz in einen int-Wert um.
 xyz.toString() wandelt den BigInteger Wert xyz in einen String-Wert um.

Zum Vergleich zweier BigInteger-Zahlen kann compareTo() und equals() verwendet werden.
```