```java
import static org.hamcrest.CoreMatchers.equalTo;

public class ZahlumwandlungTest {

    @Test
    public void testWandeleZahlUm() {

        // Erzeuge ein Objekt der Klasse Zahlumwandlung.
        Zahlumwandlung eineZahlumwandlung = new Zahlumwandlung();
        Throwable ex = assertThrows(IllegalArgumentException.class, () ->
{eineZahlumwandlung.wandeleZahlUm(0);});
        assertThat("Zahl kleiner als 1 wurde nicht korrekt verarbeitet.",
                ex.getMessage(),
                is(equalTo(String.format("Die Nummer n=%d liegt nicht im
Interval [1, 3000] liegen. ", 0))));
        ex = assertThrows(IllegalArgumentException.class, () ->
{eineZahlumwandlung.wandeleZahlUm(3001);});
        assertThat("Zahl groeßer als 3000 wurde nicht korrekt
verarbeitet.",
                ex.getMessage(),
                is(equalTo(String.format("Die Nummer n=%d liegt nicht im
Interval [1, 3000] liegen. ", 3001))));
        assertThat("Zahl 1 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(1),
                is(equalTo("I")));
        assertThat("Zahl 5 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(5),
                is(equalTo("V")));
        assertThat("Zahl 4 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(4),
                is(equalTo("IV")));

        assertThat("Zahl 7 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(7),
                is(equalTo("VII")));
        assertThat("Zahl 9 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(9),
                is(equalTo("IX")));
        assertThat("Zahl 10 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(10),
                is(equalTo("X")));
        assertThat("Zahl 11 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(11),
                is(equalTo("XI")));
        assertThat("Zahl 13 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(13),
                is(equalTo("XIII")));
        assertThat("Zahl 14 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(14),
                is(equalTo("XIV")));
        assertThat("Zahl 15 wurde nicht korrekt umgewandelt.",
                eineZahlumwandlung.wandeleZahlUm(15),
                is(equalTo("XV")));
```

```
54          assertThat("Zahl 21 wurde nicht korrekt umgewandelt.",
55                  eineZahlumwandlung.wandeleZahlUm(21),
56                  is(equalTo("XXI")));
57          assertThat("Zahl 34 wurde nicht korrekt umgewandelt.",
58                  eineZahlumwandlung.wandeleZahlUm(34),
59                  is(equalTo("XXXIV")));
60
61          assertThat("Zahl 99 wurde nicht korrekt umgewandelt.",
62                  eineZahlumwandlung.wandeleZahlUm(99),
63                  is(equalTo("XCIX")));
64          assertThat("Zahl 10 wurde nicht korrekt umgewandelt.",
65                  eineZahlumwandlung.wandeleZahlUm(10),
66                  is(equalTo("X")));
67          assertThat("Zahl 3000 wurde nicht korrekt umgewandelt.",
68                  eineZahlumwandlung.wandeleZahlUm(3000),
69                  is(equalTo("MMM")));
70          assertThat("Zahl 2500 wurde nicht korrekt umgewandelt.",
71                  eineZahlumwandlung.wandeleZahlUm(2500),
72                  is(equalTo("MMD")));
73          assertThat("Zahl 1500 wurde nicht korrekt umgewandelt.",
74                  eineZahlumwandlung.wandeleZahlUm(1500),
75                  is(equalTo("MD")));
76
77          /*
78           * Tests hier ergaenzen.
79           */
80
81      }
82  }
83
```