Bit Operations Part I

Background

You've seen how Java facilitates bit-level manipulation of data items (and presumably wrote code to reinforce your understanding) using the operators &, |, ~, <<, >>, and >>>. In this assignment you will perform compound tasks using the bit-level operations.

Specification

Create a class called **`BitOperations`** that includes the following methods:

- Extract the left 4 bits (most significant nibble) of a **`byte`**. The bits should be returned in the right 4 bits (least significant nibble) of a **`byte`** and the remaining 4 bits must be 0. Example: 10101111 -> 00001010. The method definition is

```
public static byte left(byte _byte)
```

- Extract the right 4 bits (least significant nibble) of a **`byte`**. The bits should be returned in the right 4 bits (least significant nibble) of a **`byte`** and the remaining 4 bits must be 0. Example: 10101111 -> 00001111. The method definition is

```
public static byte right(byte _byte)
```

- Extract the left 6 bits (most significant bits) of a **`byte`**. The bits should be returned in the right 6 bits (least significant bits) of a **`byte`** and the remaining 2 bits must be 0. Example: 10101111 -> 00101011. The method definition is

```
public static byte sixbits(byte _byte)
```

- Swap the most and least significant bytes in a word (**`short`**) value. Example: 1010101011111111 -> 1111111110101010. The method definition is
-

```
public static short lrswap(short _in)
```

Use the provided class **`BitOperationsTest`** (which includes a main method) to demonstrate your code.

Deliverables

- Source code (.java) file
- Reflective essay describing
    - Successes
    - Difficulties (if any) and how you addressed them
    - Lessons learned
    - Screen shot of your running program showing requested (above) results