Issues with Range Class methods

1- Contains ():

It is a Boolean function that returns true if a given value is within the range, however, it doesn't include the lower bound as one of the range elements, which, is why the function is not correct.

For Example:

```
Range rangeObj = new Range( lower: 1,  upper: 10);
Assertions.assertTrue(rangeObj.contains(1));
```

```
org.opentest4j.AssertionFailedError:
Expected :true
Actual   :false
```

2- Combine ():

It is a static Range function that returns a Range obj that is the combination of two given ranges and if one of the ranges is equal to null the result would be the other range that is not null, however, the function doesn't work properly when combining two given ranges as it always uses the lowest upper bound, which, is why the function is not correct.

For Example:

```
Assertions.assertEquals(new Range( lower: 1,  upper: 30),
        Range.combine(new Range( lower: 1,  upper: 20), new Range( lower: 21,  upper: 30)));
```

```
org.opentest4j.AssertionFailedError:
Expected :Range[1.0,30.0]
Actual   :Range[1.0,20.0]
```

3- Intersects ():

It is a Boolean function that returns true if a given range intersects with the range obj, however, it always returns true even if there was no intersection, which, is why the function is not correct.

For Example:

```
Assertions.assertFalse(rangeObj.intersects(30, 40));
```

```
org.opentest4j.AssertionFailedError:
Expected :false
Actual   :true
```

4- Expand ():

It is a static Range function that returns a Range obj that is the result of adding margins(percentages) to an existing Range, hence, extending both lower and upper bound of the range as lower bound gets lower and higher bound gets higher, however, the lower bound always gets higher, which, is why the function is not correct.

For Example:

```
Assertions.assertEquals(
        new Range( lower: 0,  upper: 6),
        Range.expand(new Range( lower: 1,  upper: 5),  lowerMargin: 0.25,  upperMargin: 0.25)
);
```

```
org.opentest4j.AssertionFailedError:
Expected :Range[0.0,6.0]
Actual   :Range[2.0,6.0]
```

4- Shift ():

This function does not have an issue the only thing different about it, is that it throws an IllegalArgumentException instead of the InvalidParameterException that is supposed to be thrown when one of the parameters is null, however, there's not much of a difference between both throws except that IllegalArgumentException is more genuine and belongs to java.lang package while the other throw belongs to java.security package.