Of course! Here are two more examples of string manipulation in Java:

Example 5: Checking for String Containment

In this example, we'll check if a string contains another string using the `contains` method.

```java
public class StringContainmentExample {
    public static void main(String[] args) {
        String mainString = "The quick brown fox jumps over the lazy dog";
        String searchString = "fox";

        // Check if the mainString contains the searchString
        boolean contains = mainString.contains(searchString);

        if (contains) {
            System.out.println("The mainString contains the searchString.");
        } else {
            System.out.println("The mainString does not contain the searchString.");
        }
    }
}
```

In this code, we have a `mainString` and a `searchString`, and we use the `contains` method to check if the `mainString` contains the `searchString`. The output will be:

```
The mainString contains the searchString.
```

Example 6: Removing Whitespace

In this example, we'll remove leading and trailing whitespace from a string using the `trim` method.

```java
public class StringTrimExample {
    public static void main(String[] args) {
        String inputString = "   This is a string with whitespace.   ";

        // Remove leading and trailing whitespace
        String trimmedString = inputString.trim();
        System.out.println("Original String: '" + inputString + "'");
        System.out.println("Trimmed String: '" + trimmedString + "'");
    }
}
```

In this code, we have an `inputString`, and we use the `trim` method to remove any leading and trailing whitespace. The output will be:

```
Original String: '   This is a string with whitespace.   '
Trimmed String: 'This is a string with whitespace.'
```

Certainly! Here are two more examples of string manipulation in Java:

Example 7: String Comparison

In this example, we'll compare two strings using the `equals` method to check if they are equal.

```java
public class StringComparisonExample {
    public static void main(String[] args) {
        String str1 = "Hello, World!";
        String str2 = "Hello, World!";
        String str3 = "Hello, Java!";

        // Compare str1 and str2
        boolean isEqual1 = str1.equals(str2);

        // Compare str1 and str3
        boolean isEqual2 = str1.equals(str3);

        System.out.println("str1 equals str2: " + isEqual1);
        System.out.println("str1 equals str3: " + isEqual2);
    }
}
```

In this code, we have three strings, `str1`, `str2`, and `str3`, and we use the `equals` method to compare them. The `equals` method returns `true` if the two strings are equal and `false` otherwise. The output will be:

```
```

```
str1 equals str2: true
str1 equals str3: false
```

Example 8: String Splitting with Regular Expressions

In this example, we'll split a string into tokens using regular expressions with the `split` method.

```java
public class StringSplitWithRegexExample {
    public static void main(String[] args) {
        String data = "John,Smith,30,New York";

        // Split the data using a comma as a delimiter
        String[] tokens = data.split(",");

        System.out.println("Data fields:");
        for (String token : tokens) {
            System.out.println(token);
        }
    }
}
```

In this code, we have a `data` string containing comma-separated values, and we use the `split` method with a regular expression (`,` in this case) as the delimiter to split the string into tokens. The output will be:

```
Data fields:
John
Smith
30
New York
```

These examples demonstrate string comparison using the `equals` method and string splitting with regular expressions using the `split` method. String manipulation is a fundamental aspect of Java programming and is essential for handling and processing textual data in various applications.