# String Class

**In Java, string is basically an object that represents sequence of char values.**

**The Java platform provides the String class to create and manipulate strings.**

**Java String class provides a lot of methods to perform operations on strings such as compare(), concat(), equals(), split(), length(), replace(), compareTo(), intern(), substring() etc.**

**The Java String is immutable which means it cannot be changed. Whenever we change any string, a new instance is created.**

**The java.lang.String class is used to create a string object.**

## There are two ways to create String object:

- **By string literal**
- **By new keyword**

## Java String literal is created by using double quotes. For Example:

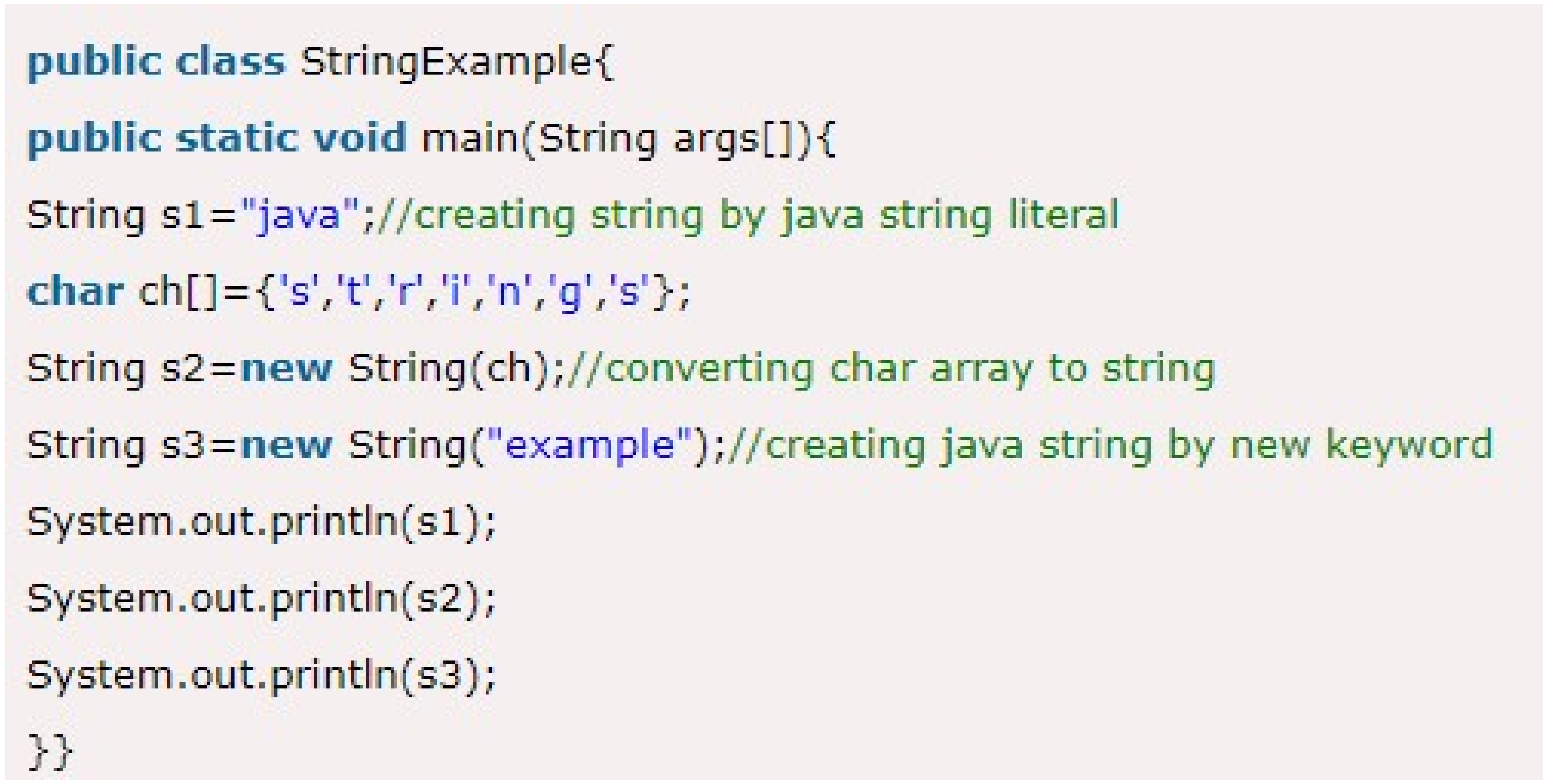
*String s="welcome";*

## By new keyword

*String s=new String("Welcome");*

*//creates two objects and one reference variable*

**In such case, JVM will create a new string object in normal (non-pool) heap memory, and the literal "Welcome" will be placed in the string constant pool. The variable s will refer to the object in a heap (non-pool).**



```
public class StringExample{
public static void main(String args[]){
String s1="java";//creating string by java string literal
char ch[]={'s','t','r','i','n','g','s'};
String s2=new String(ch);//converting char array to string
String s3=new String("example");//creating java string by new keyword
System.out.println(s1);
System.out.println(s2);
System.out.println(s3);
}}
```

## Let us see Java String class methods

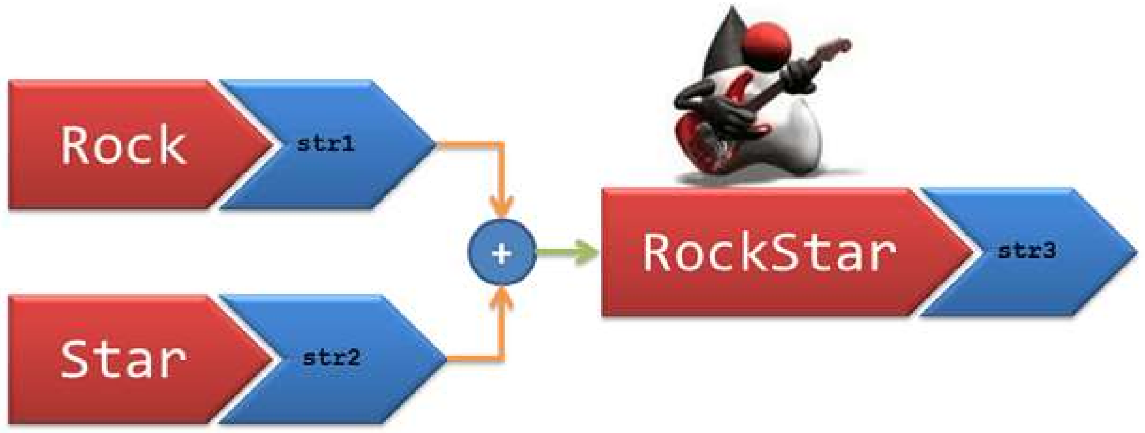
`public class String`

| | | |
|---|---|---|
| | `String(String s)` | *create a string with the same value as* `s` |
| | `String(char[] a)` | *create a string that represents the same sequence of characters as in* `a[]` |
| `int` | `length()` | *number of characters* |
| `char` | `charAt(int i)` | *the character at index* `i` |
| `String` | `substring(int i, int j)` | *characters at indices* `i` *through* (`j-1`) |
| `boolean` | `contains(String substring)` | *does this string contain* `substring`*?* |
| `boolean` | `startsWith(String prefix)` | *does this string start with* `prefix`*?* |
| `boolean` | `endsWith(String postfix)` | *does this string end with* `postfix`*?* |
| `int` | `indexOf(String pattern)` | *index of first occurrence of* `pattern` |
| `int` | `indexOf(String pattern, int i)` | *index of first occurrence of* `pattern` *after* `i` |
| `String` | `concat(String t)` | *this string, with* `t` *appended* |
| `int` | `compareTo(String t)` | *string comparison* |
| `String` | `toLowerCase()` | *this string, with lowercase letters* |
| `String` | `toUpperCase()` | *this string, with uppercase letters* |
| `String` | `replace(String a, String b)` | *this string, with* as *replaced by* bs |
| `String` | `trim()` | *this string, with leading and trailing whitespace removed* |
| `boolean` | `matches(String regexp)` | *is this string matched by the regular expression?* |
| `String[]` | `split(String delimiter)` | *strings between occurrences of* `delimiter` |
| `boolean` | `equals(Object t)` | *is this string's value the same as* `t`*'s?* |
| `int` | `hashCode()` | *an integer hash code* |

```
String a = new String("now is");
String b = new String("the time");
String c = new String(" the");
```

| *instance method call* | *return type* | *return value* |
|---|---|---|
| `a.length()` | `int` | `6` |
| `a.charAt(4)` | `char` | `'i'` |
| `a.substring(2, 5)` | `String` | `"w i"` |
| `b.startsWith("the")` | `boolean` | `true` |
| `a.indexOf("is")` | `int` | `4` |
| `a.concat(c)` | `String` | `"now is the"` |
| `b.replace("t", "T")` | `String` | `"The Time"` |
| `a.split(" ")` | `String[]` | `{ "now", "is" }` |
| `b.equals(c)` | `boolean` | `false` |

# String Concatenation:



- We have two strings str1 = “Rock” and str2 = “Star”.
- “concat” method of String class and second is using arithmetic “+” operator can be used to join the two strings to form “RockStar”.

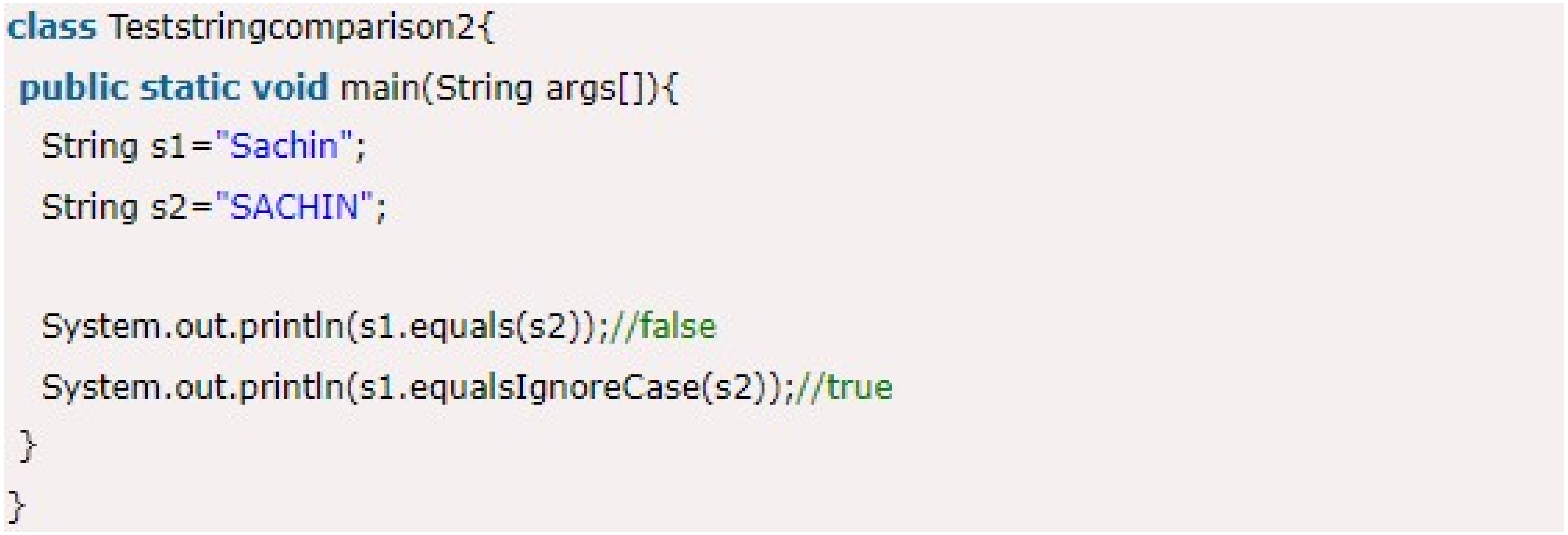
## Java String compare

**It is used in authentication (by equals() method), sorting (by compareTo() method), reference matching (by == operator) etc.**

## equals( )

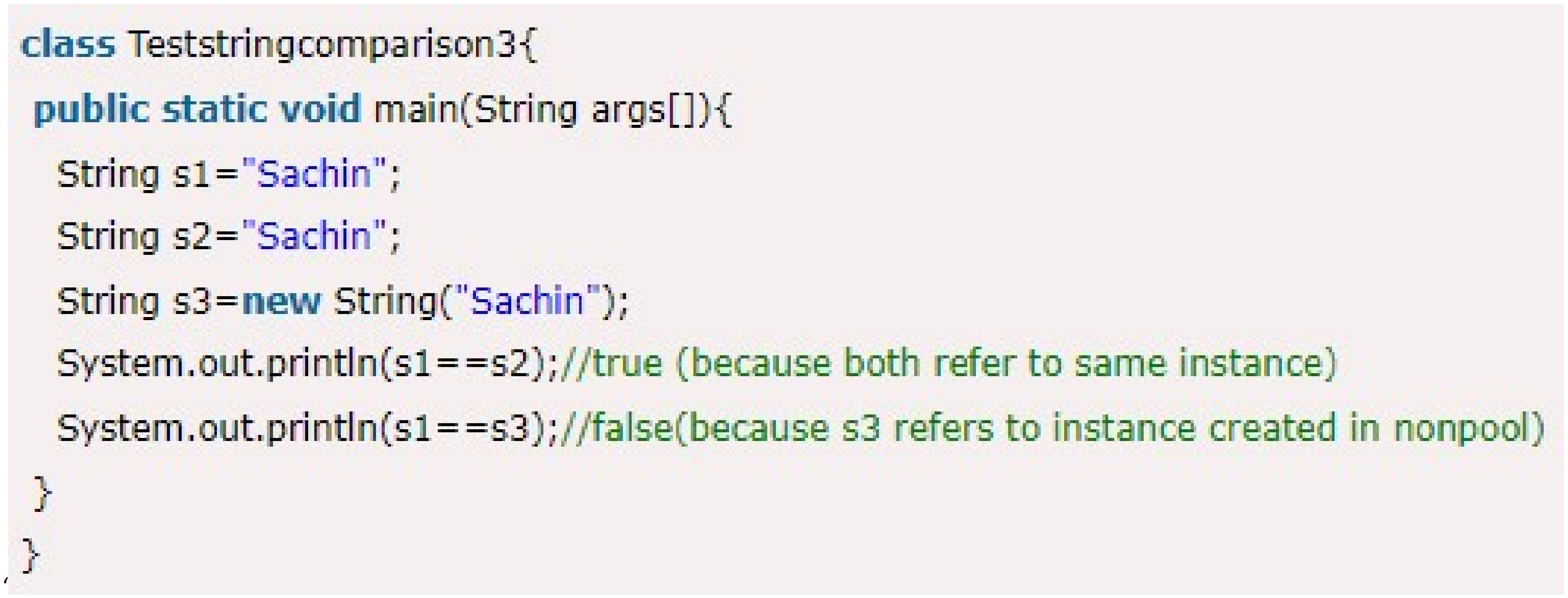
**The String equals() method compares the original content of the string.**

- **public boolean equals(Object another) compares this string to the specified object.**
- **public boolean equalsIgnoreCase(String another) compares this String to another string, ignoring case.**



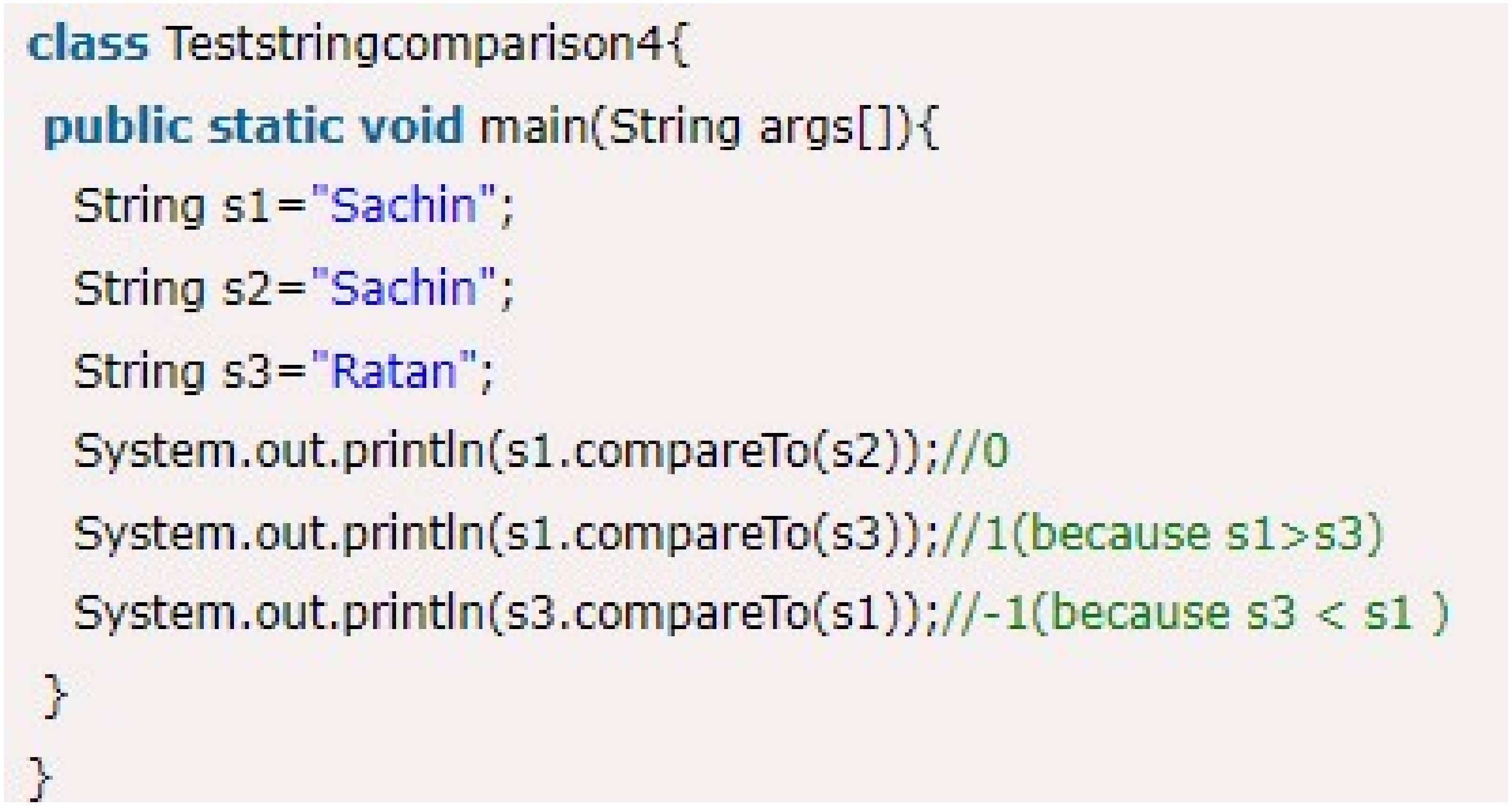
```
class Teststringcomparison2{
 public static void main(String args[]){
  String s1="Sachin";
  String s2="SACHIN";

  System.out.println(s1.equals(s2));//false
  System.out.println(s1.equalsIgnoreCase(s2));//true
 }
}
```

## String compare by == operator



```
class Teststringcomparison3{
 public static void main(String args[]){
   String s1="Sachin";
   String s2="Sachin";
   String s3=new String("Sachin");
   System.out.println(s1==s2);//true (because both refer to same instance)
   System.out.println(s1==s3);//false(because s3 refers to instance created in nonpool)
 }
}
```

## String compare by compareTo() method



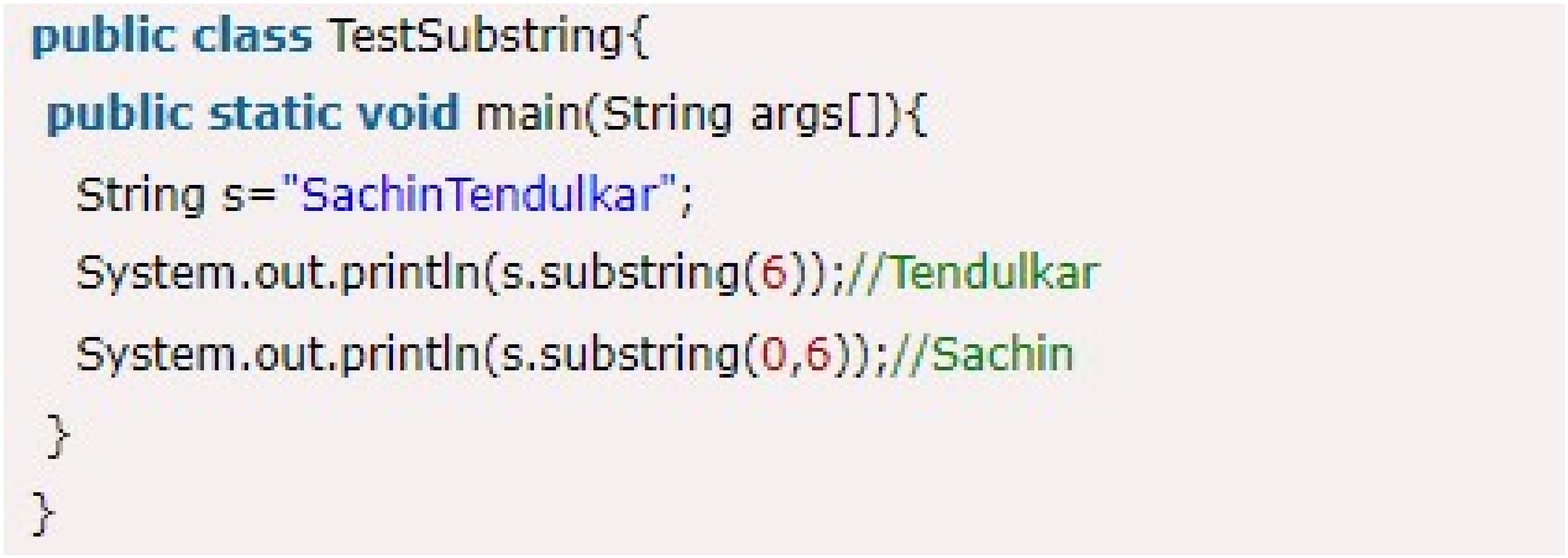
```
class Teststringcomparison4{
 public static void main(String args[]){
   String s1="Sachin";
   String s2="Sachin";
   String s3="Ratan";
   System.out.println(s1.compareTo(s2));//0
   System.out.println(s1.compareTo(s3));//1(because s1>s3)
   System.out.println(s3.compareTo(s1));//-1(because s3 < s1 )
 }
}
```

## Substring in Java

**A part of string is called substring. In other words, substring is a subset of another string. In case of substring startIndex is inclusive and endIndex is exclusive.**

1. **public String substring(int startIndex): This method returns new String object containing the substring of the given string from specified startIndex (inclusive).**

2. **public String substring(int startIndex, int endIndex): This method returns new String object containing the substring of the given string from specified startIndex to endIndex.**



```
public class TestSubstring{
 public static void main(String args[]){
   String s="SachinTendulkar";
   System.out.println(s.substring(6));//Tendulkar
   System.out.println(s.substring(0,6));//Sachin
 }
}
```

*Reference of String methods will be provided.*

## NEXT SESSION: Math Class