Object-Oriented Programming (CS F213)

Module II: Arrays and Strings in Java

CS F213 RL 8.1: Strings in Java

BITS Pilani

Dr. Pankaj Vyas
Department of Computer Science, BITS-Pilani, Pilani Campus

CS F213 RL 8.1 : Topics

- Strings in Java

String Introduction

- Represents character strings such as “a”, “David”, “Mike” etc
- Every String literal (such as “123”, “Ram”, “151103” etc.) is a String type object in Java and hence any String class method can be invoked via string type literals also. For Example, “abc”.length(), “abc”.equals(“ABC”);
- String str = “abc”; , or String str = new String(“abc”); are equivalent statements
- Strings in Java represents immutable strings
- Every string has a length which can be retrieved using length() method. For Example “Mike”.length() returns 4
- Every character of String object is idexed and index varies from 0 to L-1 (Where ‘L’ is length of String). For example, string “Mike” has character ‘M’ at index 0, character ‘i’ at index 1 etc.
- Getting characters out of index will results in StringIndexOutofBoundsException.

Important String Constructors

1. String() → Creates an Empty String, length =0
2. String(char[] str) → Creates a string from char array
3. String(char chars[],int start, int numChars) → Creates a String from chars array starting from index ‘start’ to ‘start + numChars – 1’
4. String(String str) → Creates a String from another String
5. String(byte[] bytes) → Creats a String from byte array
6. String(byte bytes[],int start, int numChars) → Creats a String from bytes starting from index ‘start’ to ‘start + numChars – 1’

String Constructors: Example

```
// Creating Strings from String Literals
String str    = "abc";                    //            String str = new String("abc");
String str1   = new String("abc");

// Creating Strings from char[] Array
char[]        chs = {'o','b','j','e','c','t'};
String str2 = new String(chs);            // str2:“object"

String str3 = new String(chs,2,4);        // start-index=1, number-of-characters=4,index range: 2 to 5 (“ject”);

//String str4 = new String(chs,3,4);      // start-index=3, number-of-characters=4,index range: 3 to 6
```

Results in StringIndexOutofBoundsException

```
  // Creating Strings from byte[] Array
  byte[]        bytes = {45,65,90,78,23,89};

  String str5 = new String(bytes);          // str5: -AZN↨Y

  String str6 = new String(bytes,2,4);      // start-index=1, number-of-characters=4,index range: 2 to 5, str6: ZN↨Y

  //String str7 = new String(bytes,3,4); // start-index=3, number-of-characters=4,index range: 3 to 6
```

Comparing Strings : equals() Method

- public boolean equals(String other) → Returns true if this string is equal to other otherwise false. Two strings are equal if they have same character at each index and their lengths are equal.
- equals() → Checks states of two string references
- Examples
  1. “abc”.equals(“abc”) returns true
  2. String s1 = “xyz”; String s2 = “abc”; s1.equals(s2) or s2.equals(s1) returns false

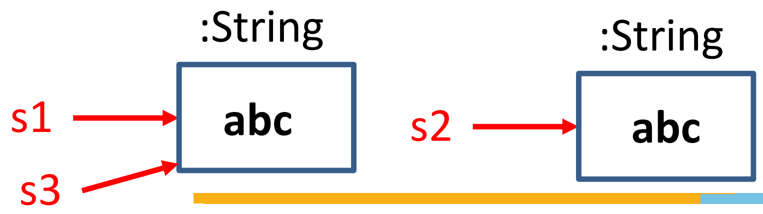
Comparing Strings : equals() Method vs ==

- Equality operator (==) returns true if both string references are pointing to same object otherwise false.
- equals() method checks the states of the string references
- Example:

```
String      s1 = new String(“abc”);
String      s2 = new String(“abc”);
String      s3 = s1;
if(s1 == s2)                    System.out.println(“Hello”);
else                            System.out.println(“Hi”);

if(s1.equals(s2))               System.out.println(“Hello”);
else                            System.out.println(“Hi”);
```

Hi

Hello

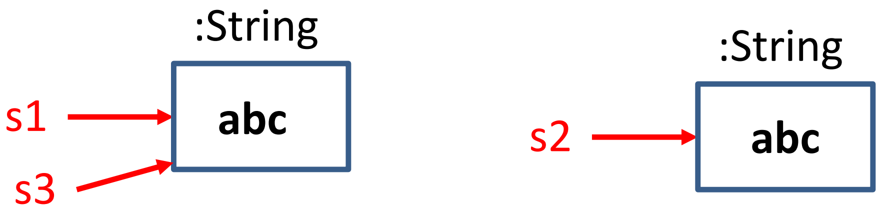


Comparing Strings : equals() Method vs == ….

- Example:

```
String      s1 = new String("abc");
String      s2 = new String("abc");
String      s3 = s1;
if(s1 == s3)                    System.out.println("Hello");
else                            System.out.println("Hi");
```

Hello

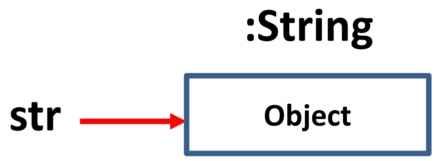


Predict the Output

```
//File Name: StringDemo.java
class StringDemo
{
        public static void main(String[] args)
        {
                String   str      =      new    String("Object");

                str               =      str +  " Oriented";

                str               =      str +  " Programming";

                System.out.println(str);

        }// End of Method
}// End of class StringDemo
```

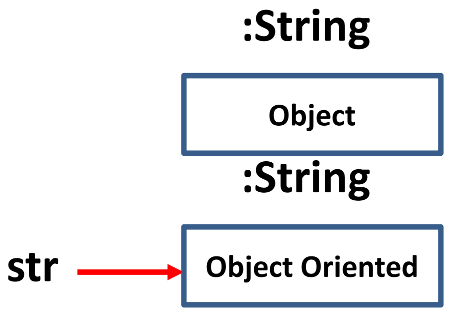
Predict the Output

```
//File Name: StringDemo.java
class StringDemo
{
        public static void main(String[] args)
        {
                String    str =     new       String("Object");

                str          =     str +     " Oriented";

                str          =     str +     " Programming";

                System.out.println(str);

        }// End of Method
}// End of class StringDemo
```



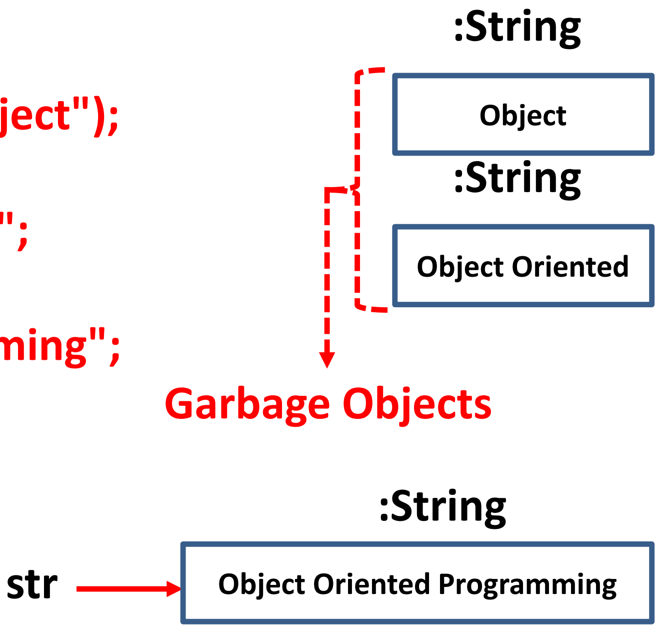
Predict the Output

```
//File Name: StringDemo.java
class StringDemo
{
        public static void main(String[] args)
        {
                String   str =    new     String("Object");

                str         =     str +   " Oriented";

                str         =     str +   " Programming";

                System.out.println(str);

        }// End of Method
}// End of class StringDemo
```



Predict the Output

```
//File Name: StringDemo.java
class StringDemo
{
	public static void main(String[] args)
	{
		String	str =	new	String("Object");

		str		=	str +	" Oriented";

		str		=	str +	" Programming";

		System.out.println(str);

	}// End of Method
}// End of class StringDemo
```



```
Object Oriented Programming
```

Each Time You Update a string reference via + operator , A new String Object is Created with updated Contents

Predict the Output

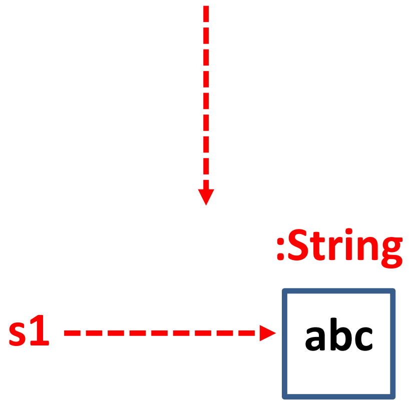
```
//File Name: StringDemo.java
class StringDemo
{
        public static void main(String[] args)
        {
                String   str      =       new      String("Object");

                str      =        new      String(“Object Oriented”);

                str      =        new      String(“Object Oriented Programming“);

                System.out.println(str);

        }// End of Method
}// End of class StringDemo
```

How Many String Objects are Created?

```
//File Name: StringDemo.java
class StringDemo
{
		public static void main(String[] args)
		{
				String s1 = "abc";

				String s2 = "abc";

				String s3 = "abc";

				String s4 = "abc";
		}// End of Method
}// End of class StringDemo
```

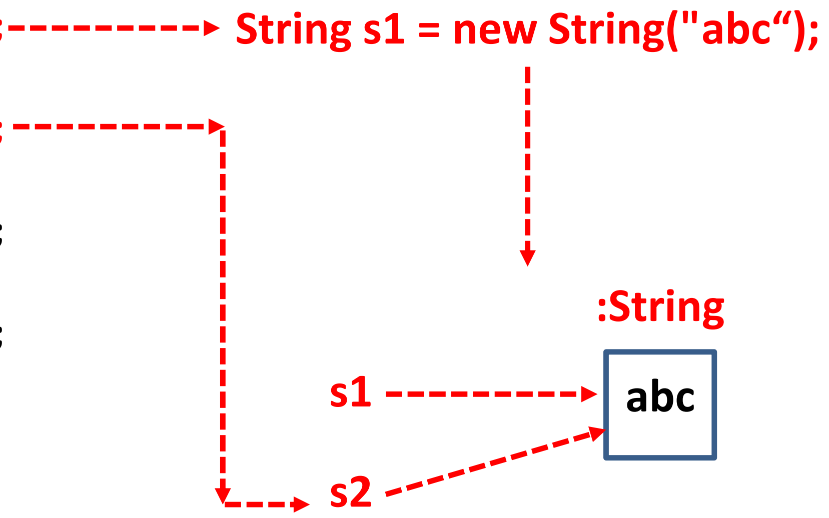
How Many String Objects are Created?

```
//File Name: StringDemo.java
class StringDemo
{
		public static void main(String[] args)
		{
				String s1 = "abc";---------► String s1 = new String("abc“);

				String s2 = "abc";

				String s3 = "abc";

				String s4 = "abc";
		}// End of Method
}// End of class StringDemo
```



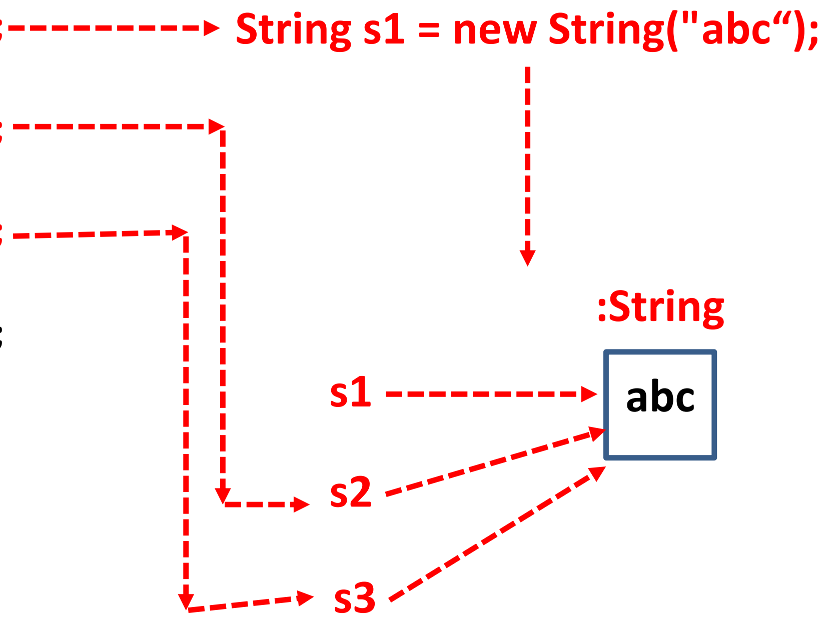
How Many String Objects are Created?

```
//File Name: StringDemo.java
class StringDemo
{
        public static void main(String[] args)
        {
                String s1 = "abc";
                String s2 = "abc";
                String s3 = "abc";
                String s4 = "abc";
        }// End of Method
}// End of class StringDemo
```



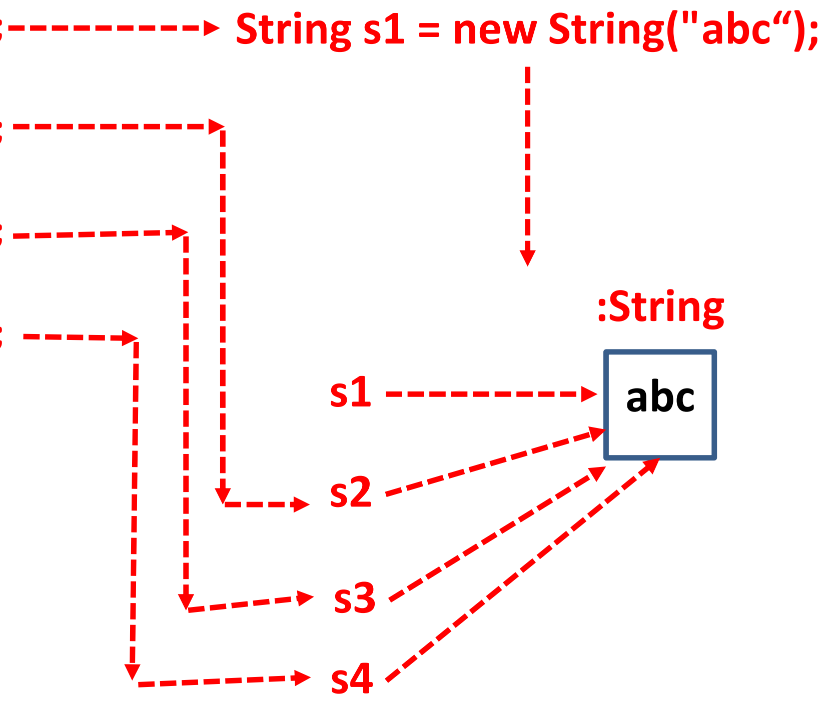
How Many String Objects are Created?

```
//File Name: StringDemo.java
class StringDemo
{
        public static void main(String[] args)
        {
                String s1 = "abc";

                String s2 = "abc";

                String s3 = "abc";

                String s4 = "abc";
        }// End of Method
}// End of class StringDemo
```



How Many String Objects are Created?

```
//File Name: StringDemo.java
class StringDemo
{
		public static void main(String[] args)
		{
			String s1 = "abc";
			String s2 = "abc";
			String s3 = "abc";
			String s4 = "abc";
		}// End of Method
}// End of class StringDemo
```



Only One Object is Created in This Example

Predict The Output?

```
//File Name: StringDemo.java
class StringDemo
{
		public static void main(String[] args)
		{
				String s1 = "a" + "b" + "c" + "d";

				String s2 = "ab" + "cd";

				String s3 = "abc" + "d";

				String s4 = "a" + "bc" + "d";

				 if(s1 == s2)			System.out.println("Hello");
				else				System.out.println("Hi");

				if(s2 == s3)			System.out.println("Hello");
				else				System.out.println("Hi");

				if(s3 == s4)			System.out.println("Hello");
				else				System.out.println("Hi");

		}// End of Method
}// End of class StringDemo
```

<<OUTPUT>>

```
F:\>java StringDemo
Hello
Hello
Hello
```

Thank You