**44-542 Object-Oriented Programming**

**Formatting and Printing Strings**

**(Using printf and String.format)**

**Method format:**  The **String** class has a method named **format** that can be used to format strings. Here is an example showing how to use the **format** method in the **String** class to create a formatted string.

**String str1 = "Hello";**

**String str2 = "Goodbye";**

**String str3 = "Hi";**

**String str4 = "Bye";**

**String str5 = String.format("%-10s %12s", str1, str2);**

**String str6 = String.format("%-10s %12s", str3, str4);**

**System.out.println(str5);**

**System.out.println(str6);**

Output for this code segment is

**Hello Goodbye**

**Hi Bye**

The **String** format statement requires a format string as the first argument. This format string is in double quotes and consists of a “rule” for formatting each of the remaining arguments. In the example given above, **str1** and **str3** are formatted using **%-10s**, meaning they should be left-justified in a field of size 10; **str2** and **str4** are formatted using **%12s**, meaning they should be right-justified in a field of length 12. Because there is a space between **%-10s** and **%12s**, there will be an additional space between **str1** and **str3** and also between **str2** and **str4**. Text other than spaces can be inserted in the format string, and it will also be duplicated. For example, if we changed the assignment statement for str5 to

**String str5 = String.format("%-10s and %12s", str1, str2);**

then the output of the first line would be

**Hello and Goodbye**

The “s” at the end of the formatting rules indicates that the corresponding argument is a string. For integer types, replace the “s” with a “d”, and for floating-point types (floats and doubles), replace the “s” with “f”.

If the width specified is too small for the value being printed, it will be expanded to the minimum length necessary for printing. For example

**String str1 = "Hello";**

**String str2 = "Goodbye";**

**String str3 = "Hi";**

**String str4 = "Bye";**

**String str5 = String.format("%-1s %1s", str1, str2);**

**String str6 = String.format("%-1s %1s", str3, str4);**

**System.out.println(str5);**

**System.out.println(str6);**

Output for this code segment is

**Hello Goodbye**

**Hi Bye**

**Using printf:** Method **format** is convenient to use if, for example, you have a method that needs to create and return a formatted string. However, if you are going to create a formatted string and then immediately print it, you can use **printf**. We can rewrite the code above as follows,using **printf**.

**String str1 = "Hello";**

**String str2 = "Goodbye";**

**String str3 = "Hi";**

**String str4 = "Bye";**

**System.out.printf("%-10s %12s", str1, str2);**

**System.out.println();**

**System.out.printf("%-10s %12s", str3, str4);**

**System.out.println();**