Formatted Output (printf)

CSE 1310 – Introduction to Computers and Programming
Vassilis Athitsos
University of Texas at Arlington

System.out.printf

```
public class example1 {
  public static void main(String[] args) {
    int days = 31;
    String month = "July";
    double temperature = 85.3;
    System.out.printf("There are %d days in %s\n", days, month);
    System.out.printf("Average temperature in %s: %f degrees\n",
            month, temperature);
  }
}
```

```
Output:

There are 31 days in July
Average temperature in July: 85.300000 degrees
```

- System.out.printf gives you an easy way to print nicer output, by combining text, variables, and other values.

System.out.printf

```
public class example1 {
  public static void main(String[] args) {
    int days = 31;
    String month = "July";
    double temperature = 85.3;
    System.out.printf("There are %d days in %s\n", days, month);
    System.out.printf("Average temperature in %s: %f degrees\n",
            month, temperature);
  }
}
```

- printf works as follows:
  - It starts printing the text in the first argument.

```
There are
```

System.out.printf

```
public class example1 {
  public static void main(String[] args) {
    int days = 31;
    String month = "July";
    double temperature = 85.3;
    System.out.printf("There are %d days in %s\n", days, month);
    System.out.printf("Average temperature in %s: %f degrees\n",
            month, temperature);
  }
}
```

- printf works as follows:
  - It starts printing the text in the first argument.
  - When it finds the first % sign, it prints the second argument.

```
There are 31
```

System.out.printf

```
public class example1 {
  public static void main(String[] args) {
    int days = 31;
    String month = "July";
    double temperature = 85.3;
    System.out.printf("There are %d days in %s\n", days, month);
    System.out.printf("Average temperature in %s: %f degrees\n",
            month, temperature);
  }
}
```

- printf works as follows:
  - It starts printing the text in the first argument.
  - When it finds the first % sign, it prints the second argument.
  - It continues printing text.

```
There are 31 days in
```

System.out.printf

```
public class example1 {
  public static void main(String[] args) {
    int days = 31;
    String month = "July";
    double temperature = 85.3;
    System.out.printf("There are %d days in %s\n", days, month);
    System.out.printf("Average temperature in %s: %f degrees\n",
                month, temperature);
  }
}
```

- printf works as follows:
  - It starts printing the text in the first argument.
  - When it finds the first % sign, it prints the second argument.
  - It continues printing text.
  - When it finds the second % sign, it prints the third argument.

```
There are 31 days in July
```

System.out.printf

```
public class example1 {
  public static void main(String[] args) {
    int days = 31;
    String month = "July";
    double temperature = 85.3;
    System.out.printf("There are %d days in %s\n", days, month);
    System.out.printf("Average temperature in %s: %f degrees\n",
            month, temperature);
  }
}
```

- printf works as follows:
  - It starts printing the text in the first argument.
  - When it finds the first % sign, it prints the second argument.
  - It continues printing text.
  - When it finds the second % sign, it prints the third argument.
  - And so on, until the entire text is processed.

System.out.printf

```
public class example1 {
  public static void main(String[] args) {
    int days = 31;
    String month = "July";
    System.out.printf("There are %d days in %s\n", days, "July");
    System.out.printf("Average temperature in %s: %f degrees\n",
            month, (85.1 + 85.5) / 2.0);
  }
}
```

- The values that you provide in the second argument, third argument, and so on, can be:
  - variables, like **days** in the example above.
  - constants, like **"July"** in the example above.
  - expressions, like **(85.1 + 85.5) / 2.0** in the example above.

Format Specifiers

```
public class example1 {
  public static void main(String[] args) {
    int days = 31;
    String month = "July";
    System.out.printf("There are %d days in %s\n", days, "July");
    System.out.printf("Average temperature in %s: %f degrees\n",
            month, (85.1 + 85.5) / 2.0);
  }
}
```

- %d, %f, %s are called **format specifiers**.
- A format specifier must match the value that will be printed.
  - %d is for values of type **int**
  - %f is for values of type **double**
  - %s is for values of type **String** or **char**
  - %c is for values of type **char**.
  - %b is for values of type **boolean**.

Specifying Width

- After the % sign, you can put a number, specifying the minimum width for that value. For example:
  - %5d means "allocate **at least** 5 spaces for that int".
  - %10s means "allocate **at least** 10 spaces for that string".
  - %7f means "allocate **at least** 7 spaces for that double".
  - %7.2f means "allocate **at least** 7 spaces for that double, but only two after the decimal point".
  - %.2f means "allocate as many spaces as needed for that double, but only two after the decimal point".
- Note the words **“at least”** in the above explanations.
  - If you specify a certain width, but the value actually needs **more** width than that in order to be displayed, it will be given the width that it is needed.
- For example, if you use %10s, but the string has 15 characters, then all 15 characters will be printed.

Specifying Width

```
public class example1 {
  public static void main(String[] args) {
    System.out.printf("%20s, current temperature: %8.2f\n",
            "Dallas", 106.7431);
    System.out.printf("%20s, current temperature: %8.2f\n",
            "San Francisco", 64.918262);
    System.out.printf("%20s, current temperature: %8.2f\n",
            "surface of the sun", 12000.0);
  }
}
```

- By specifying a width for every value, you get nicely aligned columns in the output.

Specifying Width

```
public class example1 {
  public static void main(String[] args) {
    System.out.printf("%20s, current temperature: %8.2f\n",
            "Dallas", 106.7431);
    System.out.printf("%20s, current temperature: %8.2f\n",
            "San Francisco", 64.918262);
    System.out.printf("%20s, current temperature: %8.2f\n",
            "surface of the sun", 12000.0);
  }
}
```

```
Output:

              Dallas, current temperature:   106.74
       San Francisco, current temperature:    64.92
  surface of the sun, current temperature: 12000.00
```

Not Specifying Width

```
public class example1 {
  public static void main(String[] args) {
    System.out.printf("%s, current temperature: %f\n",
            "Dallas", 106.7431);
    System.out.printf("%s, current temperature: %f\n",
            "San Francisco", 64.918262);
    System.out.printf("%s, current temperature: %f\n",
            "surface of the sun", 12000.0);
  }
}
```

```
Output:

Dallas, current temperature: 106.743100
San Francisco, current temperature: 64.918262
surface of the sun, current temperature: 12000.000000
```

- Compare the previous output to this one.
- In this version of the code, we do not specify widths in printf.
- The output does not look as nice.

Printing a New Line with \n

```
public class example1 {
  public static void main(String[] args) {
    System.out.printf("%20s, current temperature: %8.2f\n",
            "Dallas", 106.7431);
    System.out.printf("%20s, current temperature: %8.2f\n",
            "San Francisco", 64.918262);
    System.out.printf("%20s, current temperature: %8.2f\n",
            "surface of the sun", 12000.0);
  }
}
```

- When you want to print a new line, put the special code \n in your text.

Printing a New Line with \n

```
public class example1 {
  public static void main(String[] args) {
    System.out.printf("%20s, current temperature: %8.2f",
            "Dallas", 106.7431);
    System.out.printf("%20s, current temperature: %8.2f",
            "San Francisco", 64.918262);
    System.out.printf("%20s, current temperature: %8.2f",
            "surface of the sun", 12000.0);
  }
}
```

```
Output:

 Dallas, current temperature:   106.74       San Francisco,
current temperature:    64.92  surface of the sun, current
temperature:
```

- If you forget new lines, the output can look pretty ugly!

Syntax of System.out.printf

- Syntax:
  System.out.printf("$t_1 f_1 t_2 f_2 t_3 f_3 \ldots t_n f_n t_{n+1}$", v_1, v_2, v_3, …, v_n);

  - t_i is text. You can put in there whatever you want.
  - f_i is a *format specifier*. It specifies several things:
    - Value v_i should be printed at that point.
    - The type of value v_i.
    - How many characters should v_i occupy.
  - v_i is an int, double, or string.
    - It can be a variable.
    - It can be a constant, like 5, or 2.5, or "hello".
    - It can be any expression that evaluates to an int, double, or string.

Syntax of System.out.printf

- Syntax:
  System.out.printf("$t_1 f_1 t_2 f_2 t_3 f_3 \ldots t_n f_n t_{n+1}$", v_1, v_2, v_3, …, v_n);
  - t_i is text. You can put in there whatever you want.
  - f_i is a *format specifier*. It specifies several things:
  - v_i is an int, double, or string.

```
System.out.printf("There are %d days in %s\n", 31, "July");
```

- What is each t_i in the line above?

Syntax of System.out.printf

- Syntax:
  System.out.printf("$t_1 f_1 t_2 f_2 t_3 f_3 \ldots t_n f_n t_{n+1}$", v_1, v_2, v_3, ..., v_n);
  - t_i is text. You can put in there whatever you want.
  - f_i is a *format specifier*. It specifies several things:
  - v_i is an int, double, or string.

```
System.out.printf("There are %d days in %s\n", 31, "July");
```

- What is each t_i in the line above?
  - t_1 = "There are "
  - t_2 = " days in "
  - t_3 = "\n"

Syntax of System.out.printf

- Syntax:
  System.out.printf("$t_1f_1t_2f_2t_3f_3…t_nf_nt_{n+1}$", v_1, v_2, v_3, …, v_n);
  - t_i is text. You can put in there whatever you want.
  - f_i is a *format specifier*. It specifies several things:
  - v_i is an int, double, or string.

```
System.out.printf("There are %d days in %s\n", 31, "July");
```

- What is each f_i in the line above?

Syntax of System.out.printf

- Syntax:
  System.out.printf("$t_1 f_1 t_2 f_2 t_3 f_3 \ldots t_n f_n t_{n+1}$", v_1, v_2, v_3, …, v_n);
  - t_i is text. You can put in there whatever you want.
  - f_i is a *format specifier*. It specifies several things:
  - v_i is an int, double, or string.

```
System.out.printf("There are %d days in %s\n", 31, "July");
```

- What is each f_i in the line above?
  - f_1 = %d
  - f_2 = %s

Syntax of System.out.printf

- Syntax:
  System.out.printf("$t_1f_1t_2f_2t_3f_3…t_nf_nt_{n+1}$", v_1, v_2, v_3, …, v_n);
  - t_i is text. You can put in there whatever you want.
  - f_i is a *format specifier*. It specifies several things:
  - v_i is an int, double, or string.

```
System.out.printf("There are %d days in %s\n", 31, "July");
```

- What is each v_i in the line above?

Syntax of System.out.printf

- Syntax:
  System.out.printf("$t_1 f_1 t_2 f_2 t_3 f_3 \ldots t_n f_n t_{n+1}$", v_1, v_2, v_3, …, v_n);
  - t_i is text. You can put in there whatever you want.
  - f_i is a *format specifier*. It specifies several things:
  - v_i is an int, double, or string.

```
System.out.printf("There are %d days in %s\n", 31, "July");
```

- What is each v_i in the line above?
  - v_1 = 31
  - v_2 = "July"

The Circles Program, Revisited

```
import java.util.Scanner;

public class hello1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("Please enter the radius: 
");
    double radius = in.nextDouble();
    double circumference = 2 * Math.PI * radius;
    double area = Math.PI * Math.pow(radius, 2);
    System.out.println(circumference);
    System.out.println(area);
  }
}
```

<-- Last version we saw.
Used **println**.

Example Output:

The Circles Program, Revisited

```
import java.util.Scanner;

public class hello1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("Please enter the radius: ");
    double radius = in.nextDouble();
    double circumference = 2 * Math.PI * radius;
    double area = Math.PI * Math.pow(radius, 2);
    System.out.println(circumference);
    System.out.println(area);
  }
}
```

<-- Last version we saw.
Used **println**.

Example Output:

```
Please enter the radius: 10
62.83185307179586
314.1592653589793
```

The output does not look very nice.

- Too many decimals.
- No text.

Can we get output like this?

```
Please enter the radius: 10
The circumference is 62.83.
The area is 314.16.
```

The Circles Program, Revisited

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("Please enter the radius: ");
    double radius = in.nextDouble();
    double circumference = 2 * Math.PI * radius;
    double area = Math.PI * Math.pow(radius, 2);
    System.out.printf("The circumference is %.2f.\n", circumference);
    System.out.printf("The area is %.2f.\n", area);
  }
}
```

Improved version, using **printf**.
Example Output:

```
Please enter the radius: 10
The circumference is 62.83.
The area is 314.16.
```

Example: Computing Squares

- Write a program that:
  - Asks the user to enter a number.
  - Gets the number from user input.
  - Prints:
    The square of *X* is *Y*
    - where *X* is the number that the user typed,
    - and *Y* is the square of X.
  - Prints only two decimal digits.

Example: Computing Squares

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("Enter a number: ");
    double number = in.nextDouble();
    double square = Math.pow(number, 2);
    System.out.printf("The square of %.2f is %.2f\n",
number, square);
  }
}
```

```
Example Output:

Enter a number: 5
The square of 5.00 is 25.00
```

```
Example Output:

Enter a number: 2.4
The square of 2.40 is 5.76
```