



CS1101

Programming and Problem Solving

Dr. Gina Bai

Fall 2022

Logistics

- **PA01 - W, A, B, C** on **zyBook > Chap 11**
 - Due: Thursday, Jan 26, at 11:59pm
- **ZY-2B** on **zyBook > Assignments**
 - Due: Saturday, Jan 28, at 11:59pm
- We will do **live coding** today! That is, we code together!
Download the Receipt.java from Brightspace > Lecture 6, and open it in an IDE

Recap – Steps for Using a Variable

1. **Declare** a variable

- `<type> <varName>;`

2. **Initialize** the variable / **Assign** a value to the variable

- `<varName> = <value/expression>;`

3. **Use** the variable

- Use `<varName>` as part of an expression/argument

Recap – Variables

Q: What's wrong with the following code?

```
int a;  
System.out.print(a);
```

✗ Uninitialized

```
int a = 1;  
int b = 2;  
int a = a + b;  
System.out.print(a);
```

✗ Duplicated declaration of a

Improve the Receipt.java with variables

```
public class Receipt {  
    public static void main(String[] args) {  
        // Calculate total owed, assuming 7% tax and 18% tip.  
        System.out.println("Subtotal: " + (38.0 + 40.0 + 30.0));  
        System.out.println("Tax: " + ((38.0 + 40.0 + 30.0) * 0.07));  
        System.out.println("Tip: " + ((38.0 + 40.0 + 30.0) * 0.18));  
        System.out.println("Total: " + ((38.0 + 40.0 + 30.0) +  
                                         (38.0 + 40.0 + 30.0) * 0.07 +  
                                         (38.0 + 40.0 + 30.0) * 0.18));  
    }  
}
```

```
public class ReceiptWithVariables {  
    public static void main(String[] args) {  
        // Calculate total owed, assuming 7% tax and 18% tip.  
        double subtotal = 38.0 + 40.0 + 30.0;  
        double tax = subtotal * 0.07;  
        double tip = subtotal * 0.18;  
        double total = subtotal + tax + tip;  
  
        System.out.println("Subtotal: " + subtotal);  
        System.out.println("Tax: " + tax);  
        System.out.println("Tip: " + tip);  
        System.out.println("Total: " + total);  
    }  
}
```

What if I just want to keep
two digits after the decimal?

```
$ javac ReceiptWithVariables.java  
$ java ReceiptWithVariables  
Subtotal: 108.0  
Tax: 7.56000000000000005  
Tip: 19.4399999999999998  
Total: 135.0
```

Formatting Text with printf

zyBook Chap 6.2

Formatting Text with printf

A method to write a formatted string using the specified **format string** and **parameters**.

```
System.out.printf("<format string>", parameters);
```

A format string can contain **placeholders** to insert parameters:

```
int course = 1101;
```

```
System.out.println("This is CS" + course + ".");
```

OR equivalently,

```
System.out.printf("This is CS%d.\n", course);
```


The format specifiers for general, character, and numeric types have the following syntax:

`"%[argument_index$][flags][width][.precision]conversion"`

The format specifiers for general, character, and numeric types have the following syntax:

"%[argument_index\$][flags][width][.precision]conversion"

- The **required conversion** is a character indicating how the argument should be formatted. The set of valid conversions for a given argument depends on the argument's data type.

%d → integer

%f → floating point

%s → string

%c → char

The format specifiers for general, character, and numeric types have the following syntax:

"%[argument_index\$][flags][width][.precision]conversion"

- The optional argument **index** is an integer indicating the position of the argument in the argument list. The first argument is referenced by "1\$", the second by "2\$", etc.

```
public class PrintfDemo {  
    public static void main(String[] args) {  
        System.out.printf("%1$s, %2$s, and %3$s\n", "A", "B", "C");  
        System.out.printf("%2$s, %3$s, and %1$s\n", "A", "B", "C");  
        System.out.printf("%3$s, %1$s, and %2$s\n", "A", "B", "C");  
    }  
}
```

Use '\n' or '%n' for newline

```
$ javac PrintfDemo.java  
$ java PrintfDemo  
A, B, and C  
B, C, and A  
C, A, and B
```

The format specifiers for general, character, and numeric types have the following syntax:

"%[argument_index\$][flags][width][.precision]conversion"

- The optional **width** is a positive integer indicating the minimum number of characters to be written to the output.

```
public class PrintfDemo {  
    public static void main(String[] args) {  
        int a = 1;  
        System.out.printf("a is %2d\n", a);  
        System.out.printf("a is %02d\n", a);  
        System.out.printf("a is %12d\n", a);  
  
        int b = 100;  
        System.out.printf("b is %2d\n", b);  
        System.out.printf("b is %04d\n", b);  
        System.out.printf("b is %12d\n", b);  
    }  
}
```

```
$ javac PrintfDemo.java  
$ java PrintfDemo  
a is 1  
a is 01  
a is 1  
b is 100  
b is 0100  
b is 100
```

The format specifiers for general, character, and numeric types have the following syntax:

"%[argument_index\$][flags][width][.precision]conversion"

- The optional **flags** is a set of characters that modify the output format. The set of valid flags depends on the conversion.

```
public class PrintfDemo {  
    public static void main(String[] args) {  
        System.out.printf("%6s!\n", "Hi"); // right-aligned  
        System.out.printf("%-6s!\n", "Hi"); // use the '-' for left-aligned  
        System.out.printf("%1s!\n", "Hi"); // right-aligned  
    }  
}
```

```
$ javac PrintfDemo.java  
$ java PrintfDemo  
      Hi!  
Hi      !  
Hi!
```

The format specifiers for general, character, and numeric types have the following syntax:

"%[argument_index\$][flags][width][.precision]conversion"

- The optional **precision** is a non-negative decimal usually used to restrict the number of characters. The specific behavior depends on the conversion.

```
public class PrintfDemo {  
    public static void main(String[] args) {  
        double a = 1.0;  
        System.out.printf("a is %.2f\n", a);  
        System.out.printf("a is %.3f\n", a);  
  
        double b = 1.005;  
        System.out.printf("b is %.2f\n", b);  
        System.out.printf("b is %.4f\n", b);  
    }  
}
```

```
$ javac PrintfDemo.java  
$ java PrintfDemo  
a is 1.00  
a is 1.000  
b is 1.01  
b is 1.0050
```

Will be rounded

Q: Describe the content of the following format strings.

- `%Wd` **Integer, W characters wide, right-aligned**
- `%-Wd` **Integer, W characters wide, left-aligned**
- `%Wf` **Floating point, W chars wide, right-aligned**
- `%.Df` **Floating point, rounded to D digits after decimal, right-aligned**
- `%W.Df` **Floating point, W chars wide, D digits after decimal, right-aligned**
- `%-W.Df` **Floating point, W chars wide, D digits after decimal, left-aligned**

Live Coding – Improve the Receipt.java with printf

```
public class ReceiptFormatted {  
    public static void main(String[] args) {  
        // Calculate total owed, assuming 7% tax and 18% tip.  
        double subtotal = 38.0 + 40.0 + 30.0;  
        double tax = subtotal * 0.07;  
        double tip = subtotal * 0.18;  
        double total = subtotal + tax + tip;  
  
        System.out.println("Without formatting: ");  
        System.out.println("Subtotal: " + subtotal);  
        System.out.println("Tax: " + tax);  
        System.out.println("Tip: " + tip);  
        System.out.println("Total: " + total);  
  
        System.out.println("\nWith formatting: ");  
        System.out.printf(____);  
        System.out.printf(____);  
        System.out.printf(____);  
        System.out.printf(____);  
    }  
}
```

```
$ javac ReceiptFormatted.java  
$ java ReceiptFormatted  
Without formatting:  
Subtotal: 108.0  
Tax: 7.5600000000000005  
Tip: 19.439999999999998  
Total: 135.0
```

```
With formatting:  
Subtotal      $ 108.00  
Tax            $   7.56  
Tip            $  19.44  
Total          $ 135.00
```

12 spaces + 1space (OR 13 spaces)
One dollar sign
7 spaces


```

public class ReceiptFormatted {
    public static void main(String[] args) {
        // Calculate total owed, assuming 7% tax and 18% tip.
        double subtotal = 38.0 + 40.0 + 30.0;
        double tax = subtotal * 0.07;
        double tip = subtotal * 0.18;
        double total = subtotal + tax + tip;

        System.out.println("Without formatting: ");
        System.out.println("Subtotal: " + subtotal);
        System.out.println("Tax: " + tax);
        System.out.println("Tip: " + tip);
        System.out.println("Total: " + total);

        System.out.println("\nWith formatting: ");
        System.out.printf("%-12s $%.2f\n", "Subtotal", subtotal);
        System.out.printf("%-12s $%.2f\n", "Tax", tax);
        System.out.printf("%-12s $%.2f\n", "Tip", tip);
        System.out.printf("%-12s $%.2f\n", "Total", total);
    }
}

```

```

$ javac ReceiptFormatted.java
$ java ReceiptFormatted
Without formatting:
Subtotal: 108.0
Tax: 7.5600000000000005
Tip: 19.439999999999998
Total: 135.0

```

```

With formatting:
Subtotal $ 108.00
Tax      $   7.56
Tip      $  19.44
Total    $ 135.00

```

%-12s: Left aligned, 12 spaces for the 1st argument

%7.2f: Right aligned, 7 spaces for the 2nd argument, round to two digits after decimal

More Coding Practice

- Modify the program Receipt, so it
 - Prompts the user for the amount of subtotal with println
 - Reads in user input as a double
 - Calculates the tax (7%, set as a constant), tip (18%, set as a constant), and total given the subtotal
 - Prints the amount of subtotal, tax, tip, and total with printf in the format of

Subtotal \$ 108.00

Tax \$ 7.56

Tip \$ 19.44

Total \$ 135.00