To have your program print something to your screen, the easiest ways are

```
System.out.print("You have: ");  // print doesn't advance
System.out.println(42);          // println does advance
```

Both convert what was provided into a String and then writes the String to the screen. Print does not move the output cursor to the next line afterward and println does.

These print functions use as much space as needed to represent a number and will truncate after several digits if the fractional part is too long.

```
System.out.println(1.0/4.0);  // prints: 0.25
System.out.println(2.0/3.0);  // prints: 0.6666666666666666
```

If you want more control over the formatting of what you print, you could use printf instead.

Printf Basics

Here's how you print an int and a double with a space between them using printf. The \n advances the cursor to the next line.

```
System.out.printf("%d %f\n", 100, 1.0/4.0);
```

You pass printf a string in quotes, and in that string should be one or more percent symbols each followed by a format specifier. %d indicates an integer type (d stands for decimal number) and %f indicates a floating point type. After the quoted string, there must be a comma-separated list of expressions of the corresponding types. Printf prints everything in the quoted string with the format specifiers replaced by their list items. Printf does not advance the cursor to the next line, so you need to include \n whenever you want that to happen.

Printf Practice

1) Open an IDE and write a small program that prints all of the above examples.
2) Write a small program that outputs the following times-table using five printf statements, one per line. For the nine results in the table, use %d for each and in your printf's expression list put the appropriate

multiplications (eg, instead of 4 put 2*2).

```
    1  2  3
   --------
1 | 1  2  3
2 | 2  4  6
3 | 3  6  9
```

When you are printing floating point numbers you can control the number of digits that appear after the decimal point. The format specifier %.2f will print with exactly two digits after the decimal point (rounding automatically).

```
System.out.printf("%.2f\n", 2.0/3.0); // prints: 0.67
```

Also, if there is a number immediately after the %, then the value that is printed will use that many spaces (or more if needed). That means that the format specifier %6.2f will print a floating point number with two digits past the decimal and will use six spaces to do it.

```
System.out.printf("%6.2f\n", 2.0/3.0); // prints: "  0.67"
```

Finally, if you place a negative number immediately after the %, it does the same thing but places the extra spaces at the end rather than the beginning.

```
System.out.printf("%-6.2f\n", 2.0/3.0); // prints: "0.67  "
```

4) Write a small program that outputs the following division-table using five printf statements, one per line. For the nine results in the table, use %f for each, with appropriate formatting modifiers between the % and f. In your printf's expression list put the appropriate divisions (eg, instead of 1.50 put 3.0/2.0). Also, for the nine results, do not place any spaces in your string to do the spacing between numbers, instead use the format modifiers to control the printing width.

```
     1    2    3
   --------------
1 | 1.00 2.00 3.00
2 | 0.50 1.00 1.50
3 | 0.33 0.67 1.00
```