Screen Output

Output: information produced by a program.

Screen output

- displaying characters on the screen (numbers, letters, special characters, spaces etc)
- formatting: controlling the way information is displayed eg. tabs, line breaks, etc.

The `print` and `println` Methods

The two “commands” to produce output in Java are `System.out.print` and `System.out.println`. Here is the explanation of the “commands”: the `System` class contains a static field called `out` which is of type `PrintStream`. `print` and `println` are two methods of the `PrintStream` class. Therefore to find out the exact usage of these two methods, one should check the API specifications of the `PrintStream` class.

`print`:	prints characters and leaves the cursor on the same line
`println`:	prints characters and inserts a line break character; subsequent print statements will start on the next line

Examples of Signatures for the `print` and `println` methods

```
void print (double d)
void print (String s)
void println (char c)
void println (String s)
```

Displaying Literal Strings

Literal string is a series of characters in quotes

Examples:

(Note that ‘^’ represents a space on the screen)

Code	**Output**	**Notes**
`System.out.print(”Hellothere^bobby251!”);`	`Hellothere^bobby251!`	literal duplication

`// assume tab occurs every five spaces`
`System.out.print(”The\tanswer\tis\ngood”);`
`System.out.print(“This^is\n\ncorrect!”);`
	`The^^answer^^^^is`
	`goodThis^is`
	`>`
	`correct!`
		control characters (\) eg. \n, \t, \\, …

`System.out.print(“abc”);`
`System.out.println(“def”);`
	`abcdef`	print vs println

`System.out.println(“abc”);`
`System.out.println(“def”);`
	`abc`
	`def`
		print vs println

Note:

- the string is duplicated literally ie. 1 space = 1 space, no space = no space; what you see is what you get.
- notice the quotes are not written to the screen
- notice that numbers and non-letters can be part of the string
- control characters eg \t, \n,\\ can be used for formatting

Exercise:
What will the output be for the following code?

Code	**Output**
`System.out.println(“yabba^^^dabba”);`	
`System.out.println (“yes\n\nmame”);`	
`System.out.println (“\\\\\\t\t\\n2+3”);`	

`System.out.print(“big”);`
`System.out.println(“apple”);`
`System.out.println(“=New York”);`	

Escape Sequences

A character preceded by a backslash (\) is an escape sequence and has special meaning to the compiler. The following table shows the Java escape sequences:

Escape Sequence	**Description**
`\t`	tab
`\b`	backspace
`\n`	newline
`\r`	carriage return
`\f`	formfeed
`\’`	single quote
`\”`	double quote
`\\`	single backslash

Displaying numbers

Examples:

Code	**Output**	**Notes**
`System.out.println (5);`	5	integers
`System.out.println (25);`	25	integers
`System.out.println (34.789);`	34.789	real
`System.out.println (4*3);`	12	evaluates expression and prints answer
`System.out.println (8+2-3);`	7	evaluates expression and prints answer

Note:

- a number is evaluated not duplicated

Exercise:
What will the output be for the following code?

Code	Output

`System.out.println (55.6);`
`System.out.println (39.2);`
`System.out.println (85 * 3);`
`System.out.println (8+6+9-2);`	

Displaying a combination of information
To print several pieces of data in one print statement, use the + operator to concatenate (join) the characters. Note that the + sign between two numbers with act as addition.

<u>Examples:</u>

Code	Output
`System.out.println (5 + "85" + 24.2 + "\n5*2");`	58524.2 5*2
`System.out.println(“5” + “2”);`	52
`System.out.println(5 + 2);`	7
`System.out.println ("8*8=^" + 8 + 8);`	8*8=^88
`System.out.println(8 + 8 + “=8*8”);`	16=8*8