Transcript: Character Strings

>> In this set of notes we're gonna take a look at data and expressions. We'll begin with character strings and then talk about escape sequences, variables and assignment, primitive data, of which there are eight types, and then we'll take a look at if and if-else statements with simple Boolean expressions.

This will allow us to write some more interesting programs early. And then we'll look at arithmetic expressions and operator precedence, and then how we accept input from the user through the keyboard. And then we'll finish up with data conversions. So, let's take a look at character strings. Recall that a literal, a string, just a string of characters, is represented by putting double quotes around it.

Here are some examples here. "This is a string literal." "Pat Doe, 123 Main Street" and "7". Notice that "7" here is a string rather than an integer literal that you can do arithmetic with. When your program is running, every character string that you views is actually an object in Java defined by the String class.

And we'll take a look at objects later, and this will be the first class that we actually look at. So let's look at the println method again. We, we've talked about that earlier. Recall it just prints a character string, and then it advances to the next line. That's what println means, it prints out a line, so to speak, and then goes to the next line.

It's based on the System.out object, which is an output stream, and it corresponds to standard output, or stdout, we just call it standard out, which is usually the computer screen. So looking at a simple statement here, System.out.println ("War Eagle from the Auburn Plains!");. We've got the object, System.out, and we're calling the method println.

And then the parameter arg-, our argument that we're sending it is the string "War Eagle from the Auburn Plains!". Now when you see a, a, a Q, up at the top here, representing a question, this is Q1, remember to look in the Questions folder, and review those. The answers are, are there as well, in a separate folder, but remember to take a look at those.

You could probably pause and look at them now. So, let's talk about the print method as opposed to println. The print method, for system.out, similar to println, except that it does not advance to the next line after it prints. It actually, sorta leaves you, on the same line, if you will.

Transcript: Character Strings

Let's look at a first example. So this is CountOff. Class CountOff, it's got a main method, of course, and then a number of print statements. There's four println, three are print, and, and one is, it finishes up with a println. So let's step through this in the debugger.

I've set a break point here on line 12 and I'm gonna click the Debug button. So we're back to do a println down. And notice down here we printed out Down and then we ended up on the next line. So I'm going to print, execute the next statement there.

Same thing happened there. We printed out One dot dot dot and went to the next line. Two, Three, and now on line 16 in our program, I'm about to execute a, a print statement rather than println, so it's not gonna advance to the next line. Notice down here in the, Run I/O tab, it just left us, where the print left off.

And now I'm gonna execute the next line, so we got four, and then five, and then finally on the next statement, it's another println that, notice, it's gonna pick up right where we leave off. But, since it's a println, it'll actually take us to the next line. Okay?

So it, it printed and, and. And then the cursor, so to speak, where we're gonna start printing next, if we were, is on the next line there. And one more step and we end. So, you need to notice, note the diff-, difference, between println and print. Just remember println prints wherever it happens to be and then goes to the next line.

Whereas print prints wherever the next print should be in the line, but stays on that line. All right, let's look at string concatenation. We use the + operator. If it appears, in conjunction with a string, and another string, or another operand, we're gonna do concatenation. And so here we've got "Peanut butter ", that's a string, in quotes, + "and jelly".

That we just concaten those, concatenate those together, and we'd end up with peanut butter and jelly. Recall we can't break a string literal across, two lines in a program. So you'll need to break up the string if it's long and, and, use this concatenation or + operator to, to take care of that.

Transcript: Character Strings

Now, we can also append a number to a string. And let's look at some at examples of that in ConcatenationExample1. So, here we've got a main method and we're gonna print out a few strings. As you can see here on line 15, Auburn University, and then we're gonna say, + "HISTORY OF NAME CHANGES".

So that's gonna print all of that out. And then we're going to, add some years to it, when some other things are found. In fact, let's just step through this. So in this first statement here, when I step, it prints out AUBURN UNIVERSITY HISTORY OF NAME CHANGES, down below.

And now we're going to print "East Alabama Male College (founded): ", followed by "1856". And notice that "1856" on line 19 is a string, rather than an integer literal, so down below we printed that out. And, next we're gonna print out "Agriculture and Mechanical " + "College of Alabama: " + 1872);.

And notice, 1872 is not a string, it's actually an integer literal. But the fact that we're putting it together with this + and a string means that we're actually gonna do concatenation. So we get out, just the 1872 there. Now on this next line, on line 24, "Alabama Polytechnic Institute:" + (1899));.

Notice I put parentheses around that, like we might be doing some arithmetic. But it's just gonna evaluate that 1899 and actually concatenate it, as you see. And the same thing happens with "Auburn University: " + (1960));. Okay? So, just to review here, + is a binary operator. And, and when it's applied to two operands and one of them happens to be a string, then you're gonna do string concatenation.

And we know, from, from life experience that, if it's between two numbers, as in these two integer literals, 5 and 10, we would actually add into arithmetic. Now, the + operators are evaluated left to right, but you can use parentheses to, force a particular order, and so on.

So, let's, let's take a look at an example, our second example of concatenation. And here, same program except down below I've broken up the the dates here into a couple things where I'm doing addition. So let's just run in debug mode, and we'll run down here. And here we're printing out, we just printed out, Agricultural and Mechanical College of Alabama: 1872.

And now, next, is "Alabama Polytechnic Institute: " and notice I've got 1872 + 27. Well, I really wanna add those two numbers, and get 1899, but that's not what's gonna happen here because I've got a string, + 1872, and I'm evaluating these pluses from left to right. So it's actually gonna concatenate 1872 with this string, "Alabama Polytechnic Institute: And then I'll have a new string.

And I'll say + 27 and it'll concatenate that on to the end. Not exactly what one would probably want, but let's look at it. So down here in the I/O tab, we see we've got 187227. Now, what we really should have done is put parentheses around the two numbers we wanted to add, and so it would have added them, and then done the concatenation.

So here we, we've (1899 + 61)); and that'll give us, 1960, like we want, okay? So, one other item, we've got this idea of interactions in jGRASP. And, we can experiment with some string expressions there. It's this tab, the fourth tab there down below, or next to Run I/O interactions, and I'm going to just key in a string here.

The number is, I can get number out there, is. And, I'm gonna end that string, and I'm gonna say +, 10. And let's say + 20. So I'm trying to get 30 out there, and no, no semicolons. So this just evaluates the expression. And notice I get, notice I get the number is 1020.

If I add parentheses around the 10 + 20. I get the number I want there, where it says the number is 30. Now, we'll use interactions a good bit as, as we go. We can, actually enter any Java statement in that you'd put inside of a method. It'll execute.

If you just enter an expression, with no semicolon, it'll evaluate that expression and you'll get to see the result right there. Otherwise, you'll see things maybe over in the Variables tab.