Transcript: Break and Continue Statements

>> The break and continue statements are useful when dealing with loops. A break statement in a loop will skip the rest of the code in that iteration and exit the loop, so you end up beyond the loop, if you will. A continue statement will skip the rest of the code in that iteration and attempt the next iteration of the loop.

So the first one break actually breaks out of the loop, and you continue beyond the loop. Whereas the continue just goes to the next iteration, but it allows you to skip the last part of the loop, if you will. Now break and continue are statements are usually used in conjunction with an if statement inside a loop.

That is if some condition is met, you break out of a loop, or if some condition is met, you continue the loop. So let's look at a couple of examples, one is break while example, the other one is a continue while example, and again, these are in your examples folder.

So, here is the break while example, and, in these example we're gonna look at a string, called searchMe. And searchMe is set to how much wood could a woodchuck if a woodchuck could chuck wood. Now we wanna look for the last character here or, or rather the last h.

So we're gonna have a care search for last and, and we're gonna set that to h, meaning we wanna look at the last h in this string here. So to do that, we set the index to searchMe.length, searchMe is the name of the string, and we set the index to the length of the string minus 1.

So we're gonna be looking at the character d there, if you will. And then we, set boolean foundIt to false. And then we enter the loop. And so, well, since we're searching from high to low, so to speak, the end of it toward the beginning, we say index greater or equal to zero.

All that's true. We ask here if searchMe.charAt(index) equals searchForLast. searchForLast is the h. So we're just checking to see if that char, at in, at wherever the index is is equal to an h. If it is, we set foundIt to true, and we break. In other words, we get outta the loop and go down here, and, and print some things out.

So the idea is we don't continue through the loop once we find it. So a break statement is very useful with searches. If you're searching for the first item or the last item once you find it you break out of the loop and, and continue on. So let's run this in in the canvas.

Transcript: Break and Continue Statements

So here's our we're about to create searchMe. And so let's create that. And so here we see search me, and it's how much wood could a woodchuck chuck, and so on. And and then we set the character we're searching for to h. And there's, end x is set to 68, so that's down toward the end and we set found it to false.

So I'm going to just move down here so we can see 68 here there's 68 and there's index. And so we're going to now, step through this. So we're looking to see if, searchMe.charAt, the index, 68 here, is equal to h. And it's not, and so we skip that, if statement.

And we, decrement index, and we go back up and we keep going through here. We're looking for an h, and you see index is moving across here. H is gonna be at 60, by the way. So let's continue. But we're going through the loop, and here comes h.

All right, this should be h. Index is still greater than or equal to 0. And so we check to see if it is h. It is, and so we go inside the if, foundIt is set to true. And then next we're gonna break out of the loop. So instead of continuing all the way down to 0 there, we're gonna go and break out of the loop, and found it is true.

Recall we did find it. And so we would print the result out as found last edge index 60 and int, OK? Now let's look at the the next example here, a continue, statement. So this is a, a while statement, it has a continue in it. We've got the same string there, and in this case we, we're going to count the number of Cs and, and we've got a lot of Cs in here.

And we're gonna do this in, a bit of a backwards way just so we can see the continue. So, while we're in the loop, we're asking if it's not equal to C. If it's not equal to C, we're gonna go ahead and increment and then, continue so we'll go back to the top of the loop.

And skip this business down here. If we find a C, then we'll go down here and say the number Cs goes up by one, and, and we'll increment as well. And so that, that's kind of what, what we're doing here, just to show how continue works. So let's run this in a canvas.

Got our canvas, and let's begin searching, we'll just search slowly through the first couple here, so we can see. So we've got index here set to 1. And now we're searching and notice we're, looking for not equal to C. If it's not equal to C, which it's not here, we're going to, go and continue.

So this is gonna loop back to the top of the wall. Notice we skip the rest of the loop, we're continuing it. And we step, step, and we're zipping along here, and eventually, we're gonna find a C here. So we're coming up on one. 6 should be a C.

And so in this case, when we're asking not equal, that's gonna be, faults. And we're gonna go down and, actually increment number of Cs there. So that gets incremented and then we go on. So we can just, march on through this. I'm gonna go ahead and, I'm gonna move this over so we can sorta see it move through here and count the Cs.

I'll just let it play, so here it goes. And you can watch the number of Cs increment. Here comes a C right there at 14 and it went up. And we got another C coming up right here at 26. It should increment, and, when it hits there you see it go down and get the C, there's one at 32, 35.

So we're just continuing until we get to the, next C here, coming up at 47. There's 47 and we jump down out of it. We did so at 50 as well. And at 53. Got a lot of C here. And we just jump down out of it again there.

And that's gonna just end the loop right there. So a lot of processing goes on with these loops. And this just shows you how you might wanna continue a loop. In other words, ignore the last part of the loop. Or you may want to just break out of a loop.

Now just running these, let's run these both here. In this one we're looking for the H. If you just run it, this runs very fast so there's found last agent 60. So all these loops do, do a lot of work. Here's counting all the Cs, so we're gonna go through this entire list of string here looking for Cs, and it actually goes pretty fast, OK?

So just keep in mind, loops are a good way to do a lot of work, and combined with if statements, you can do some write some powerful programs.