Week7 lecture5.1

Welcome to Loops and Iteration.

Basically, this is where computers do repetitive tasks that we

humans don't want to do.

This is where the real power and the real benefit of computers happen.

Each time through the loop we have to do something smart.

But we can tell a computer to do it a million times really, really quite easily.

And so the basic idea is that at some point in the code, you go back up, right.

The idea is that you've done something, let's go do that thing again.

And the way we express that in Python is with a couple of keywords.

One of the keywords is the while keyword and the for keyword.

And so we'll start talking about the while.

And in a bit, we'll talk about definite loops using for.

These are called indefinite loops.

So while is a keyword and this is like an if statement, this is a question.

Is n greater than 0?

That's leads to a true or false answer.

And if it is true, this code executes and if it is not true, the code is skipped.

And in that respect, it is exactly the same as an if.

You could say if n was greater than 0, run this code, otherwise skip this code.

But that's where things get a little bit different.

从 :1:20 开始播放视频并学习脚本1:20

So if it's true in this case, n is 5, so it is true.

So it runs this.

It prints out 5, which prints that out and then subtracts 1 from n, which makes it 4.

But with the while statement, it goes back up again.

从 :1:33 开始播放视频并学习脚本1:33

And it rechecks the question.

That's another way to say this is here, re-checks the question.

And as long as n greater than 0 remains true, it will dive in and do this again.

So it dives in and prints out 4, and then it subtracts to 3 and goes up.

Checks again, checks again as long as it's there, down it goes again, so

it prints 3 and then subtracts to 2, and then prints 2 and subtracts to 1,

and then prints 1 and subtracts to 0.

So in this case, n right now, is we printed that and now n is 0,

and it comes up.

n is 0, and it comes up.

Is n greater than 0?

Well, it now switched from all the times yes to now no, okay?

And so, when it's no, it comes down here and it's done.

So when it comes up here, it skips down to the next line and then it finishes,

so it prints out Blastoff!

And we see that, when we came out of the loop, n was 0.

So that was like a residual value of this little variable n after the last iteration

through the loop.

And that's because it was 1 and then we subtracted 1 from it.

And that got us to 0.

从 :2:44 开始播放视频并学习脚本2:44

So, an important part of any loop is what we call the iteration variable, okay?

And that is something that changes,

because if we don't change anything in the loop, then it's going to run forever.

That's what we call an infinite loop.

So here we have n greater than 5 and we say while n is greater than 0,

which is true, print this, come back up, check again. Is n greater than 0? Yeah.

Print this, check again, is n greater than 0? Print this.

And this is when you read your shampoo and it says lather, rinse, and repeat.

This is what a computer scientist thinks when they read that.

It's like, that would be an infinite loop.

You're going to run out of water,

you're going to run out of shampoo, you're going to run out of something, right?

But the technical problem is, is that n is not changing.

n is not changing.

And so what happens is there's no way for this true to become false.

There's no way, it just stays true, and so that's an infinite loop.

And that will literally run until your computer's battery runs out or

you unplug it or you hit some escape or whatever.

If your computer is spinning in a beach ball, it's got some little weird

little thing, it's probably in a loop that is either infinite or nearly infinite.

So this is a bad thing to do because it locks up computers.

If you write code to do this and you run it on your computer,

you will probably get some kind of a spinning beach ball or

whatever that your computer, whichever your operating system does,

because it's not good to put your computer in an infinite loop.

从 :4:8 开始播放视频并学习脚本4:08

And so we need somehow to have an iteration variable that changes and

in this previous one, I subtracted 1 from n.

And so this variable eventually went from true, true, true, true,

true, false, and so it became false.

This question became false, which gets us out of the loop.

This variable that we use to control it is what we call an iteration variable.

Because it controls how long the iterations run and

when the iterations stop.

So this is an infinite loop because we're not affecting the iteration variable.

The iteration variable still is n.

And then we have another sort of version of this loop that shows that these loops

are what are called zero-trip loop.

And that is when this one comes in, this one starts out false.

And so it skips it. And so this basically functions like an if statement.

So it's okay, it comes in, it's false, skip.

Never runs either of these statements.

从 :5:7 开始播放视频并学习脚本5:07

Now, we have some statements that we can use to get out of a loop.

One of them is the break statement.

And it's an executable statement.

When it runs, it basically breaks out of the loop,

moves to the line beyond the end of the loop.

从 :5:25 开始播放视频并学习脚本5:25

So wherever we're at, when this code runs, it jumps out of the loop.

从 :5:32 开始播放视频并学习脚本5:32

And so we're going to have a loop.

This is an infinite loop.

I just told you not to do infinite loops,

but we're going to use break to get out of this loop.

So while true, remember I said, it starts out true, can it ever get to false?

Well, in this case, it's never going to get to false because it's always true.

While true, we've constructed an infinite loop and what does this loop do?

It asks for a line of input with a little arrow sign.

And then we type something like hello there.

It goes in, we check to see if the line we typed was d-o-n-e.

If so, we break, that's not true, so then we print the line.

Then we go up, prompt again, print it.

从 :6:10 开始播放视频并学习脚本6:10

Prompt again and this time, we type in done.

So done gets typed.

So this becomes true, so we execute here.

And it immediately leaves, so it doesn't print the word done here.

It prints that string to indicate that we've left.

This can be quite a long loop in here.

This can be a lot of and you can even say break a couple of different places.

And you usually put the break inside an if test.

And as soon as the break executes, the loop is done, it is out of the loop.

So you can think of it as

从 :6:44 开始播放视频并学习脚本6:44

Here's the block, here's the de-indent.

So it's this much, this is a block.

从 :6:48 开始播放视频并学习脚本6:48

The break escapes the block, right?

It gets out of that loop block, the block of code that is the loop.

从 :6:56 开始播放视频并学习脚本6:56

You can think of it as like a transporter, right?

You're coming in, you're doing a bunch of stuff.

You're doing this and you're doing this and you're doing this.

Some other thing, some other thing, and like, oh bang!

Hit the break key!

Leave. Right?

So whatever it is,

however much code is in this loop, the break says get out of this loop.

Get to the next line beyond the end of the loop.

It's not to go to a different place in the loop.

It's escape the loop directly and instantaneously.

从 :7:25 开始播放视频并学习脚本7:25

That's the break statement.

The other statement that does something like this is the continue.

And so continue basically says quit on the current iteration and

go to the next iteration.

So it skips out of the loop, but

it doesn't skip to the line beyond it, it skips back up to the top.

So continue says, oh, we're going to go up to the top.

Break says get out and continue says don't do the rest of this iteration but

go up and do the next iteration.

So we have taken this exact same bit of code and

the only bit we added to it was right here.

So, we read a line.

从 :8:3 开始播放视频并学习脚本8:03

If the first character is a pound sign, we continue.

If the entire line is done, we break.

And so breaking is what gets us out at this point.

Get out and we say done.

从 :8:14 开始播放视频并学习脚本8:14

And what we do is if we find a line like this one that has pound sign as the first

character, we skip back to the beginning of the loop.

We go back down and we do another input.

So we don't print that.

But we say something here and so it does not print the line.

So it skips that.

So it's like skipping to the top of the loop, not skipping out of the loop.

So break skips out of the loop and continue skips to the top of the loop.

从 :8:41 开始播放视频并学习脚本8:41

And again,

same kind of thing, you're running through this loop, all kinds of ways.

You don't hit the continue.

You just happen to stumble and you run a continue statement and you go back up to

the top of the loop, meaning there's nothing down here that you run.

从 :8:55 开始播放视频并学习脚本8:55

The continue doesn't get you out of the loop,

the continue goes to the next iteration basically.

Abandons the current iteration and goes to the next iteration.

So these while loops are kind of indefinite.

You can construct them cleverly.

They just go and go and go until some logic or condition becomes false.

Hopefully, you've constructed all the loops except one.

I constructed it, so that it would eventually either exit the loop or

the condition would become false.

从 :9:23 开始播放视频并学习脚本9:23

It's a little hard to sometimes verify that these things are perfect.

从 :9:27 开始播放视频并学习脚本9:27

Sometimes it's a little tricky to make sure that the loop will terminate.

Up next, we'll switch from using while to using for and talk about definite loops.