Finding a Gene in DNA

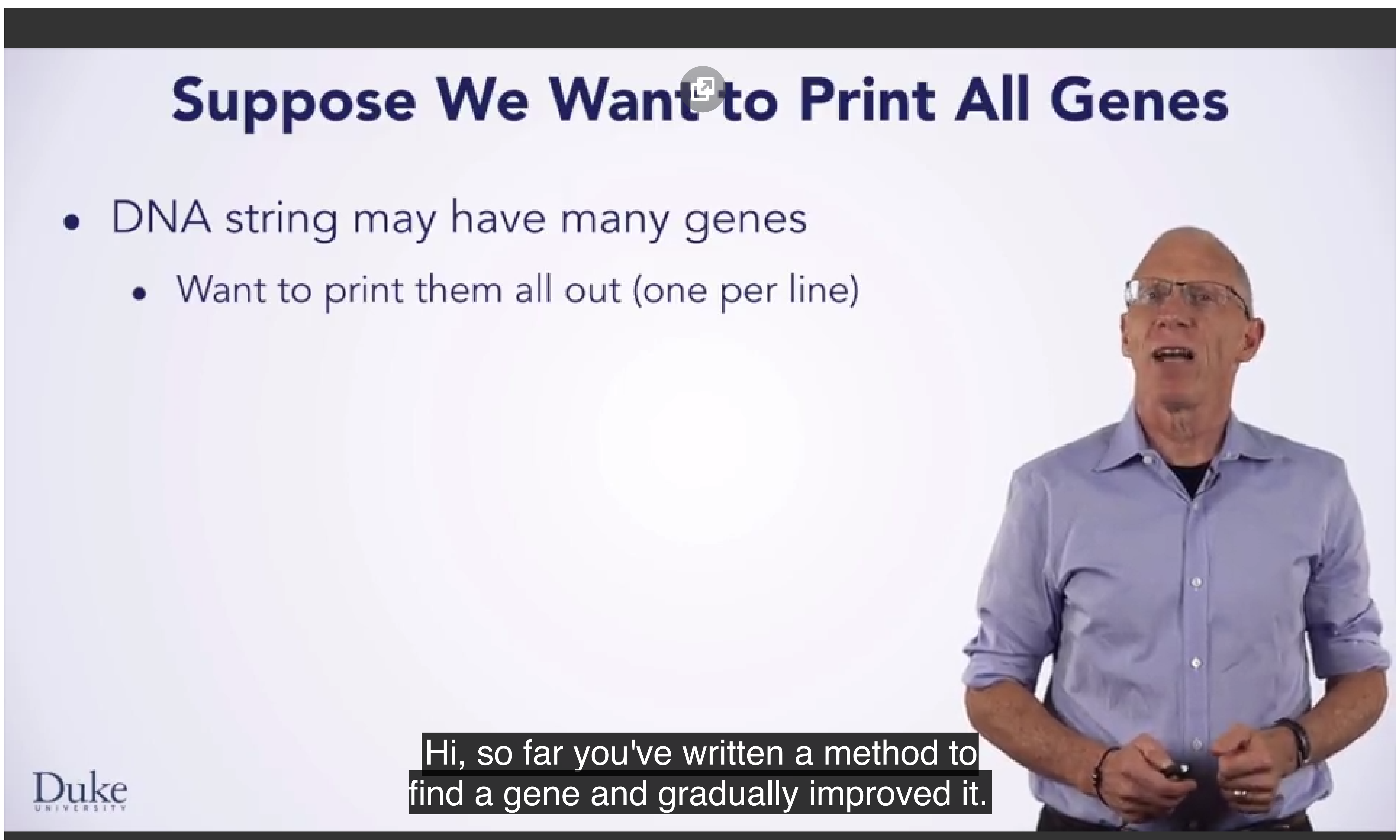
Finding All Genes in DNA

Introduction	48 sec
Conceptual Understanding	4 min
While Loops	9 min
While Loop Syntax and Semantics	3 min
Coding While Loops	6 min
Three Stop Codons	5 min
Coding Three Stop Codons - Part I	7 min
Coding Three Stop Codons - Part II	4 min
Logical And / Or	8 min
Coding And / Or	6 min
Finding Multiple Genes	5 min
Translating to Code	8 min
Programming Exercise: Finding Many Genes	10 min
Practice Quiz: Finding All Genes in DNA	4 questions

Debugging Code

Using the StorageResource Class

Review



Suppose We Want to Print All Genes

- DNA string may have many genes
  - Want to print them all out (one per line)

Duke

Hi, so far you've written a method to find a gene and gradually improved it.

Finding Multiple Genes

Have a question? Discuss this lecture in the week forums.

Downloads

Lecture Video mp4

Subtitles (English) WebVTT

Transcript (English) txt

Would you like to help us translate the transcript and subtitles into additional languages?

Interactive Transcript

Search Transcript English

0:03
Hi, so far you've written a method to find a gene and gradually improved it. It's gotten much more sophisticated than when you first started. Although it's still a bit of a simplication of what you'd actually do. Rather than continuing to refine this method, let's ponder a different facet of searching for genes. So far, you've only been looking for the first gene in the string. However, strings may have many genes. What if you wanted to find them all, and print them all out? You can already find one. Although you might want to make some slight adjustments to the methods so you can start looking midway through the string. Since you can find one and you want to find many, you want to repeat steps using a loop. Loops should be getting pretty familiar by now. Since you might want to repeat things, as long as there are more genes, you can make use of the while loop that you recently learned about. However, there's a bit of a difficulty here. We wont know if there are more genes until we start searching for them. This may make it seem hard to write a loop condition.

1:16
There are many ways to deal with this situation in code. But, the one we're going to teach you about is how to use a break statement to leave the middle of the body of a loop.

1:25
We'll work through a slightly shorter example than we normally walk through in developing the algorithm to print all genes.

1:33
If you don't understand this completely when we finish, pause the video and work through steps 1, 2, and 3 yourself.

1:41
We're going to show you how it operates on this DNA string as we write it. First, we're going to set startIndex to 0. StartIndex will represent where we start looking for the next gene. Then we're going to repeat some steps, as long as there are more genes after startIndex. We want to find the next gene after startIndex. We want to print that gene out. And then set startIndex to just past the end of the gene we found. To show you how the algorithm would continue working on the string, we then go back to Step 2 and keep repeating these steps, as there are more genes after startIndex. Notice, this is the difficulty we alluded to before. We need to know if we will find more genes or not, but we haven't looked for them yet.

2:36
We'd find the next gene, print it out, update startIndex, and then realize we should stop repeating steps because there are no more genes.

2:48
However, we have this difficulty. We need to know if we have more genes in Step 2, but we don't look for a gene until Step 3, which makes the algorithm a bit awkward to implement. Really, we'd like to make a decision about whether to keep going or to stop right here, between Steps 3 and 4.

3:11
Here's a slightly modified version of the algorithm which does exactly that. Notice that our repetition instructions no longer have any condition, they just say repeat these steps. We'll figure out later when to stop. Likewise, we now have another step in the middle of the loop that says, leave the loop if we did not find any genes.

3:35
This would be much easier to implement,

3:39
once we learn how to translate these kind of steps into code.

3:44
For step two, repeating steps without checking any particular condition, you can simply write, while true. If the condition of a while loop is simply true, the code will always enter the loop body when we reach the top, because true always evaluates to true. The other new piece of Java syntax we'll need is the break statement. This is how you say, leave this loop.

4:10
In this example, we've translated if no gene was found, into an if statement that says, if gene.isEmpty. The isEmpty method for a string returns true if the string is the empty string and false otherwise. Remember, our gene finding method returns the empty string whenever it can't find a gene.

4:34
Inside the if statement, we've seen leave this loop, translated into a break statement. A break statement which is written in Java simply with the keyword break, followed by a semi-colon, causes Java to leave the current loop. Regardless of whether it's a while loop or a for loop or any kind of loop you might yet learn about, like a do while loop. Basically, Java jumps past the close right curly brace that ends the loop. Now we know how to implement these steps. Let's turn the algorithm into code and try it out. We'll find every kind of gene there is.

5:14
Have fun.