

Finding a Gene in DNA

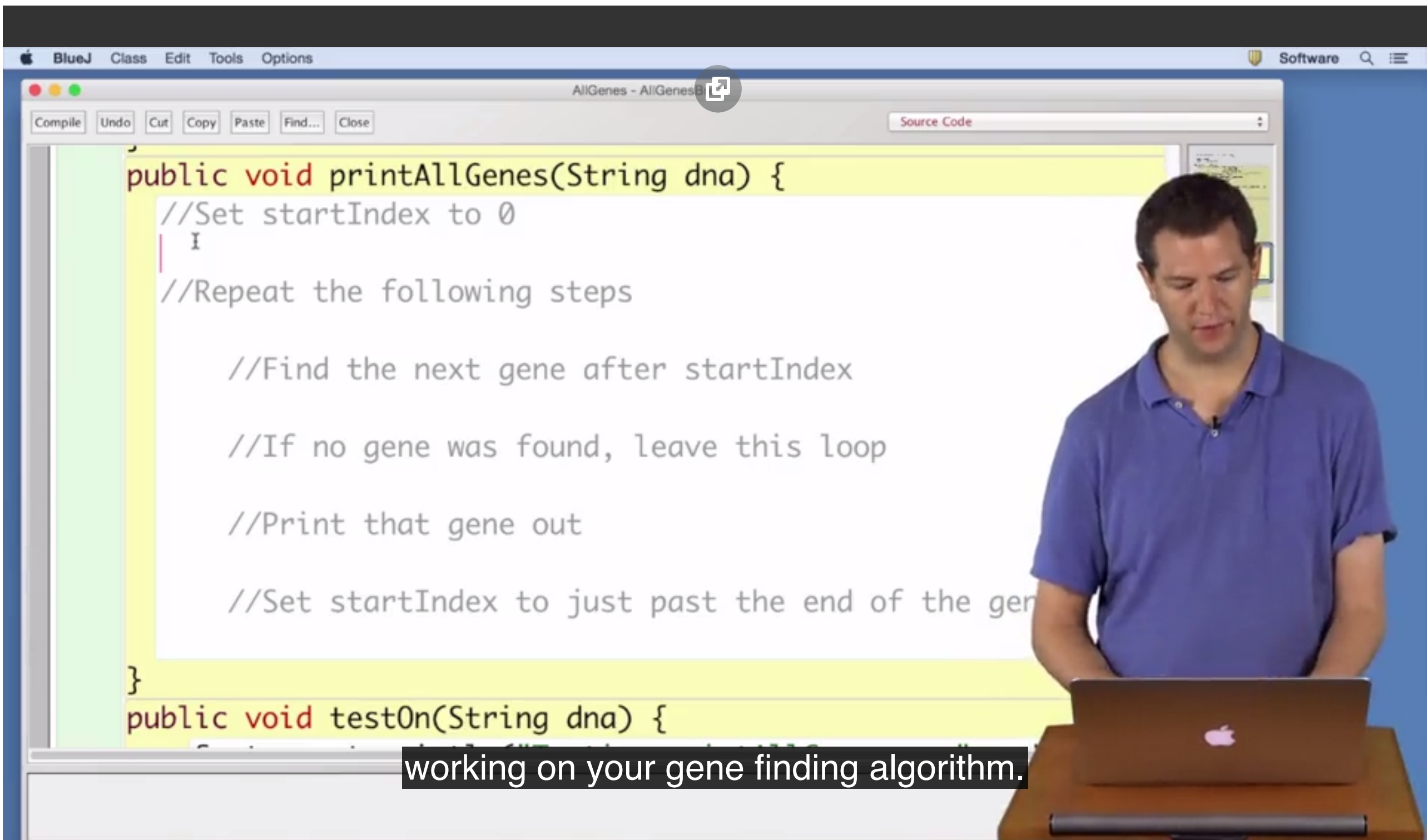
Finding All Genes in DNA

| | |
|--|-------------|
| 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



```
public void printAllGenes(String dna) {
    //Set startIndex to 0
    int i = 0;
    //Repeat the following steps

    //Find the next gene after startIndex

    //If no gene was found, leave this loop

    //Print that gene out

    //Set startIndex to just past the end of the gene
}

public void testOn(String dna) {
```

working on your gene finding algorithm.

Translating to Code

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

Interactive Transcript

Search Transcript

English

0:03
All right, so you've been learning about while loops and break statements and [working on your gene finding algorithm](#). And your current task is to write a method which will print out all the genes in a sequence of DNA.

0:15
So we have here the print all genes method which takes in a string for the DNA and has the algorithm or pseudo code that we came up with in a previous video.

0:26
Now we have earlier here in this code the gene finding algorithm that you developed along with Owen in a previous video. And we're just going to make one small change to it. I'm going to have it take one more parameter to say where to start and this is going to let us start looking for a gene somewhere in the middle of the sequence of DNA. So once we've found one at the beginning we can start looking for one after that, by passing this index, and we'll just pass that to index of the teller where to start looking for the first start code on.

1:01
So with that small change, we're going to go back here and start translating our algorithm into code.

1:07
The first thing we said to do was to set the startIndex to 0. So we haven't made a variable called start index yet, so the first thing we need to do is declare startIndex, it's going to be an int since we're setting it to 0. It's going to tell us the position of the string, also a good clue that it's an integer.

1:27
And then we're going to repeat the following steps. So you're used to repeating steps by now. You've done things for each pixel or each of a variety of other things. For each as a lot, you've also seen some examples of while, where you've had a condition for as long as something. But here we want to repeat some steps, and then figure out somewhere else inside those steps when we're done, so we're going to write while true, and then put curly braces around our steps. And then we're going to go in here and we're going to start doing stuff, and then figure out when we want to stop repeating these.

2:04
Now we want to find the next gene after startIndex. Finding a gene is a big complicated step. You've been devoting lots of effort and thought into how to do that. Fortunately we've abstracted that out into a method, so we can just call it. And let findGene that we've already written do all of that work for us. So what does findGene take? It takes a string for the dna and int saying where to start, and then it returns a string. So here we want to call findGene passing in what dna? Well, this dna that we're working with. And where do we want to start?

2:38
At startIndex.

2:40
And then it's going to give us back a string as its answer, so we want to give that a name. I'm going to call it currentGene. We didn't say to give it a name in our steps, but we kept referring to it without a name. We kind of know what we mean, but it's good to give things names in your steps anyways. And now we can refer to that gene that we found.

3:01
If no gene was found, leave this loop. So if something is the case, do some other steps. By now, you should be getting very familiar with if statements. So if no gene was found, how do we know if no gene was found? What does findGene do if there's nothing left? If you don't remember, we can go back up here and look. You can see that it returns this empty string.

3:24
So we want to know, is currentGene the empty string?

3:28
Fortunately for us, strings have a .isEmpty method, which will tell us if they're the empty string. How would you do that if you didn't know? Well, you could go look at the Java documentation. Are there other ways to figure out if a string is empty? Sure, you could see if its length was zero for example. Any way that you can find that works is a good way to do it.

3:49
All right, so if this Gene is empty, if current Gene is empty and we didn't find it, what do we want to do? We want to leave this loop, that is, we want Java's executing along and it gets here. We wanted to jump down here pass this loop, that's exactly what a break statement does, which you just learned about. So we would get here and then come down here, we basically put the loop condition in the middle.

4:14
All right, if we don't break out to that loop, we want to print out that gene, System.out.println, hopefully you're getting somewhat familiar with that.

4:23
We can print out that currentGene and then we want to update statIndex to be just past the end of the Gene. All right, so now this seems a little tricky. We wrote a step that's a little bit complicated. Maybe we have to think it through a little bit. Let's just go down here to where I've got some testing and think about what we are going to do.

4:45
I've written some methods to help us test, and let's think through one of these cases here. So this testOn just prints out that we're testing it and then it does printAllGenes.

4:58
So, we say okay, start index is going to be zero. We're going to find this gene right here, which is length nine, right? And so then after that we would want our new start index to be nine, okay? So we might think we just want to add the length of the gene that got us from 0 to 9. That's a good first guess. Now lets think about this next one. So then, we're going to go, and we're going to find this next gene here, which is length 3, 6, 9, 12, 15. But that didn't start right where we left off, right? So if we were to only add 15, we'd end up here in the middle of this gene, and we want to end up over here. So what do we need to do? Well, we need to add the length to the start position of this gene. How can we find that gene? Well, index() is becoming your friend, right? So startIndex is going to be dna.indexOf(CurrentGene, startIndex) + currentGene.length. That seems kind of magical, let me just explain that again really quickly. All right, we want to find where that current gene was in the string, we can do that by looking for it again starting at start index. So that's going to tell me, this gene was right here. And then we can add its length to end up past here. That'll work for this first one because looking for this first gene from the start index of zero is just going to give me zero. And when I add its length, I'm going to end up here, all right? So that still doesn't make sense to work it out on your own.

6:51
So, I'm now going to hit Compile.

6:55
All is well there and now I'm going to come over here and I'm going to make a new object and then I'm going to run my test method. And, you can see it says, all right, I'm testing print all genes on this sequence of DNA, and it's got two of these. Back over in my code, I tried to make this a little easier for us to see. So we've got this gene here, and then we've got this gene here, and these little vs are just for me to keep track of the codons. And those were what it printed here. I tested it on the empty string, I wouldn't expect it to find any genes in there. But it's always good to test on these wacky corner cases where maybe our code would break if we weren't careful. We didn't find any genes, but our code didn't crash either, that's really good. And then on this one, where I have this one long gene here, which we found and printed out. And then this really short gene here which we also found and printed out.

7:50
So that worked well and the important thing here, the new lesson was this while true, figure out if we want to keep going in the middle and then break.

Downloads

| | |
|----------------------|--------|
| Lecture Video | mp4 |
| Subtitles (English) | WebVTT |
| Transcript (English) | txt |

Would you like to [help us translate](#) the transcript and subtitles into additional languages?