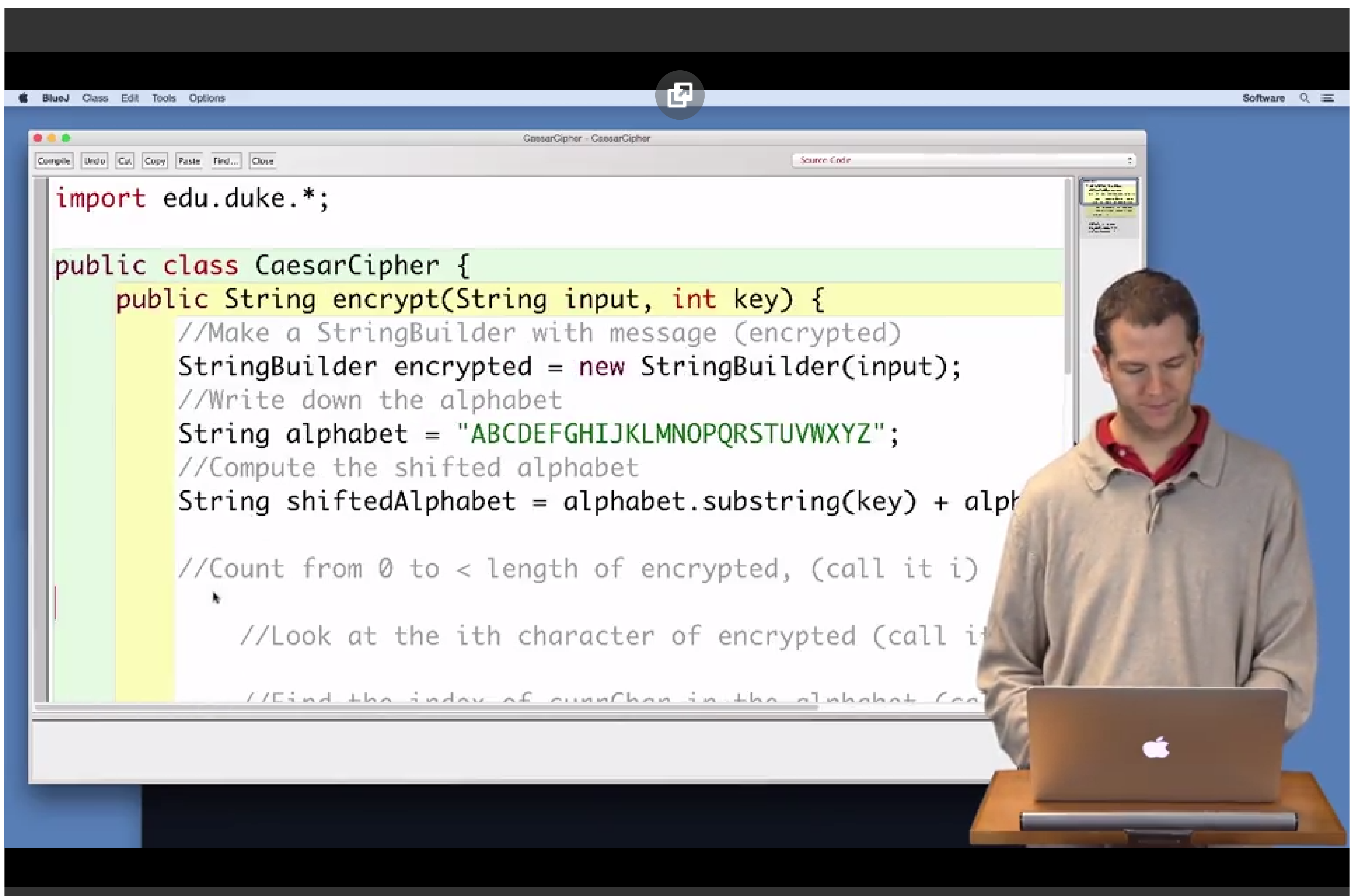
BlueJ Class Edit Tools Options
CaesarCipher - CaesarCipher
Compile Undo Cut Copy Paste Find... Close
Source Code
Software

```
import edu.duke.*;

public class CaesarCipher {
    public String encrypt(String input, int key) {
        //Make a StringBuilder with message (encrypted)
        StringBuilder encrypted = new StringBuilder(input);
        //Write down the alphabet
        String alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        //Compute the shifted alphabet
        String shiftedAlphabet = alphabet.substring(key) + alph

        //Count from 0 to < length of encrypted, (call it i)

            //Look at the ith character of encrypted (call i
```

Translating into Code

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

Interactive Transcript

Search Transcript English

0:03
Okay, now that you've developed the algorithm for Caesar Cipher, it's time to turn it into code. We've started here with the CaesarCipher class, which has an encrypt method which takes the input to encrypt, and an integer for the key to use to encrypt it. I've written here the comments, with the pseudo code that we just developed in our algorithm. And I've gone ahead and written the first two lines of code. The first step is to make a new StringBuilder with the input. And call it encrypted. And the second is to write down the alphabet, which is a string.

0:35
The next thing is to compute the shifted alphabet. So we're going to make a string, shiftedAlphabet. And remember, we saw in previous videos, that we can use substring to do that. lphabet.substring(key) +, that's what plus means for strings, alphabet.substring(0,key). That will slice the alphabet up into two pieces and can concatenate them back together to give us the shifted alphabet we want.

1:08
Now we can count from zero to being less than the length of encrypted. And we want to call each number that we count i. So this is going to be a counting for loop, for (int i = 0 i < encrypted.length(), the length of encrypted, i++). And what do we want to do? Well, we want to look at the ith character, and call it currChar. char currChar = encrypted.charAt(i). Then we want to find the index of currChar in the alphabet. int idx, we said over here call it idx, = alphabet.indexOf(currChar), and then we want to see if that's in the alphabet. As you've seen before, indexOf will return a negative one if something is not there and some other number if it is. So if would not be there if idx is index, or is negative one. We want != -1 to mean it is there.

2:19
Now we want the idxth char of shifted alphabet, and we want to call that newChar = shiftedAlphabet.charAt(idx). And now I want to replace the ith character of encrypted with newChar so encrypted.setCharAt and then I want the ith character to be newChar.

2:51
That's everything I want to do inside of that if, otherwise, do nothing. I don't need to write an alt if I don't want to do anything otherwise.

2:59
This curly braces closes my counting for loop, and at the end here, I said I want my answer, so I want to return something, to be this string inside of encrypted, so encrypted.toString.

3:15
I'm gonna compile that, it says no syntax errors. And I've written here already this method testCaesar, which is going to make a new file resource, so it'll prompt us for a file. Read it all in is a string encrypt it. It will print out the encrypted message, and then it will also decrypt it and print out the decrypted message, because if we just saw a jumble of random characters, we wouldn't really be able to tell if we did it right. But being able to tell that we got the original back makes us more confident that what we did was correct.

3:48
I'm gonna go over here to BlueJ. I'm gonna say new CaesarCipher.

3:54
I'm gonna say Test Caesar, and it's gonna prompt me for a file. And I have here this message in a file called message1.txt that says, free cake in the conference room. Maybe I want to send this to Robert without Owen seeing it and being able to get all the cake, and so I'm gonna choose message1.txt. And we can see the encrypted message here, WIVV, etc., etc. It's been made more difficult to read. And then we can see down here, where it decrypted it correctly. So I'd be a little more confident in this. I'd want to go through and check by hand that this actually came out with the correct key before I'd be completely certain that this test case worked correctly.

Downloads

Lecture Video mp4

Subtitles (English) WebVTT

Transcript (English) txt

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