Introduction to the Course

Implementing the Caesar Cipher

- Module Learning Outcomes / Resources 10 min
- A Brief History of Cryptography 5 min
- Introduction 5 min
- Creating and Manipulating Strings 5 min
- Counting Loops 9 min
- Character Class 5 min
- Developing an Algorithm 5 min
- Translating into Code 4 min
- Testing and Debugging 1 min
- Summary 40 sec
- Programming Exercise: Implementing the Caesar Cipher 10 min
- **Practice Quiz:** Implementing the Caesar Cipher 6 questions

Breaking the Caesar Cipher

Object Oriented Caesar Cipher

Review

How would you generalize the following code, which creates a String variable named **encr** that is a shifted alphabet for encryption, to be used for any value of **key**?

```
String encr = alphabet.substring(23);
encr = encr + alphabet.substring(0, 23);
```

```
String encr = alphabet.substring(23);
encr = encr + alphabet.substring(0, key);
```

```
String encr = alphabet.substring(key);
encr = encr + alphabet.substring(0, key);
```

Correct

```
String encr = alphabet.substring(key);
encr = encr + alphabet.substring(0, 23);
```

Continue

2:23 / 5:48

Creating and Manipulating Strings

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

Interactive Transcript

Search Transcript English

0:03
Welcome back. Now that you know a little bit about how a Caesar Cipher works, it is time to expand your Java knowledge so you can implement one. You should know something about strings already. If you took our previous course, Java Programming, Solving Problems with Software, you learned how to manipulate strings. If you are unfamiliar with strings, we recommend you brush up on them before proceeding.

0:25
However, to perform encryption, you need to do something with strings that you did not do in the previous course. In that course, you analyzed the contents of strings and operated on pieces of existing strings. However, you did not build up new strings. If you carefully reexamine what you saw about how a Caeser cipher works, you will see that you are making a new string by adding one character at a time. One way that you can build up a string is with concatenation, which is a fancy word for sticking things together. In Java, you can perform string concatenation with the plus operator, whenever at least one operand is a string. If one operand is a string but the other is not, than the concatenation operator figures out the string representation of the non-string operand and concatenates that.

1:12
For example, if you were to write these two strings with the plus operator between them, you would be telling Java to perform string concatenation. The result would be the string you see here, which was formed by sticking these strings together, exactly what concatenation means.

1:30
To illustrate the usefulness of this operation, think about the Caesar Cipher you're working on. You might want to take the original alphabet, as a string, and make a rearranged alphabet based on the key. You could do that by taking two pieces with substring, which we're already familiar with, and concatenating them together.

1:50
First, you would take the substring starting at 23. Remember the substring with only one argument returns the substring starting at the specified position and going all the way to the end of the string.

2:01
Then you would concatenate the sub string from zero to 23 onto the end of that string. Now you have two strings which describe the mapping from each plain text letter to the appropriate cipher text letter. We did this for the key of 23, but you should think for a moment about how you would generalize this for other keys.

2:24
As you learn more about manipulating strings, it is important to know that they are immutable. You cannot change an existing string once it has been created. Instead, if you want a different string, you must make a new one with that change. This concept may seem a bit confusing and subtle, so it helps to see it with a picture.

2:42
Here we have declared a string and initialized it to hello. Nothing new or surprising here. A common practice is to draw a picture of the effect of this statement by drawing a box for the variable s, and making an arrow pointing to the letters that make up the string hello.

3:00
If you declare another string x, and initialize it with s, you would have a picture that looks like this. Both x and s refer to the same string.

3:11
Now suppose you did s = s + World. That is you compute s concatenated with the string, space World. You are not changing the existing String, but rather making another String, which involves, copying the existing Strings like this. Notice that x is still Hello, cause you did not change the existing String, you made a different String and assigned it to s.

3:43
If you do a lot of string concatenation operations, on large strings especially, the required copying can be quite slow and inefficient. Even though we are not terribly focused on the most efficient way to do things yet, it is still a good practice to develop good habits.

3:59
If you are building large strings by adding many small pieces, you wanna use a different approach.

4:07
In fact, Java has a StringBuilder class specifically for this purpose. It provides a mutable sequence of characters. Meaning, it is like a string but you can modify it, changing and inserting characters in an efficient way.

4:20
When you create a new StringBuilder you can pass a string in to specify its initial contents. There are also many useful methods. We will name just a few of the most important ones, but recommend you consult the API documentation for a full list and more details. One useful method is append, which lets you put a string on the end. You could also pass in other types of data, which will be converted to a string before being added to the end.

4:47
You could insert a string or the string representation of other types of data at any location you want. You can get or set individual characters by their index, the numerical location where they are in the sequence of characters. When you are done manipulating the string buffer, you will often want to use toString to get the string you have made.

5:09
As before, it helps to see a picture of how these methods operate. Here we have started by creating a string builder and passing in the string Hello. We have drawn this picture with sb having an arrow pointing to the sequence of characters in the string builder.

5:23
Now if you call sb.append and pass in world you will modify the existing sequence of characters. Notice how we changed the existing sequence rather than copying them into a new sequence. You can also insert or put the characters into the middle, which would still modify the same sequence of characters like this. Great. Now you know how to build up strings from smaller pieces.

Downloads

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

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