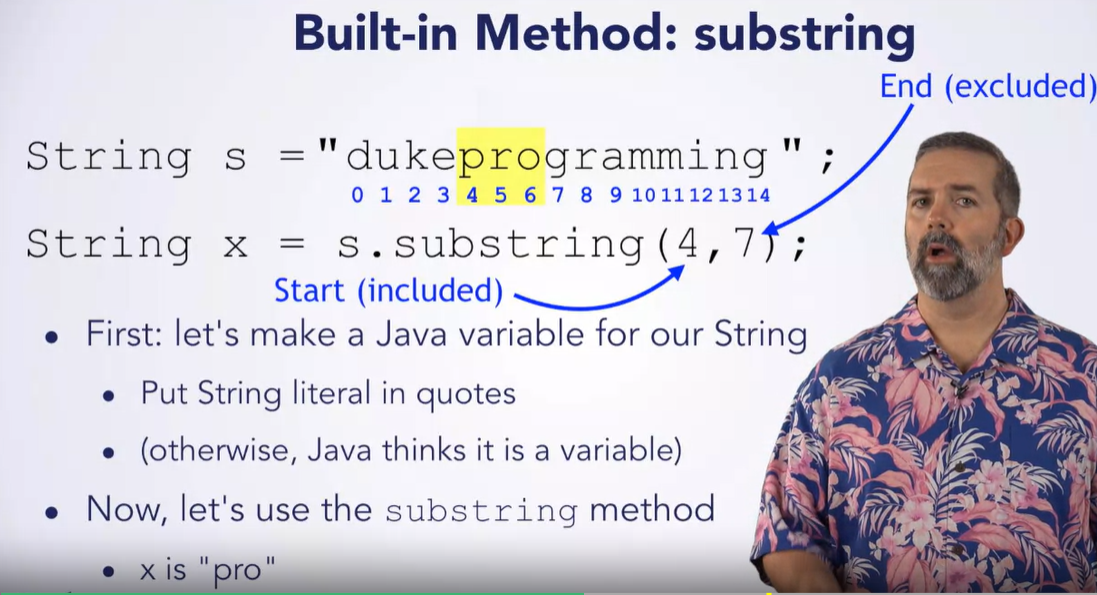
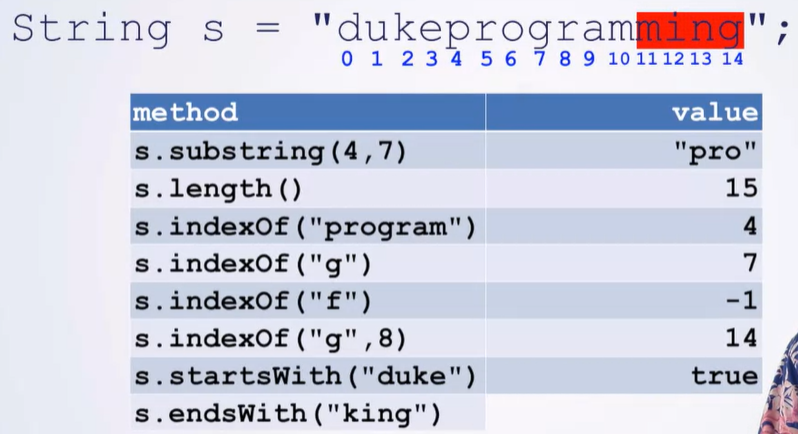
**String**

El CSV significa, valores separados por comas. Escribirás código en una lección posterior para procesar archivos CSV.





Parte 1: Encontrar un gen - Usar el algoritmo simplificado

Esta tarea es escribir el código de la lección desde cero siguiendo los pasos a continuación. Esto lo ayudará a ver si realmente entendió cómo armar el código y podría identificar una parte que no entendió completamente. Si te quedas atascado, puedes volver y ver los videos de codificación que acompañan a esta lección nuevamente. Le recomendamos que pruebe esto con muchos de los futuros ejemplos de codificación Java antes de comenzar los ejercicios de programación.

Específicamente, debe hacer lo siguiente:

1. Cree un nuevo proyecto Java llamado StringsFirstAssignments. Puede poner todas las clases para este ejercicio de programación en este proyecto.

2. Cree una nueva clase Java llamada Part1. Los siguientes métodos van en esta clase.

3. Escriba el método findSimpleGene que tiene un parámetro de cadena de ADN, que representa una cadena de ADN. Este método hace lo siguiente:

Encuentra la posición del índice del codón de inicio "ATG". Si no hay "ATG", devuelva la cadena vacía.

Encuentra la posición del índice del primer codón de parada "TAA" que aparece después del "ATG" que se encontró. Si no existe tal "TAA", devuelva la cadena vacía.

Si la longitud de la subcadena entre el "ATG" y el "TAA" es un múltiplo de 3, entonces devuelva la subcadena que comienza con ese "ATG" y termina con ese "TAA".

4. Escriba el método vacío testSimpleGene que no tiene parámetros. Debes crear cinco cadenas de ADN. Las cadenas deben tener casos de prueba específicos, tales como: ADN sin "ATG", ADN sin "TAA", ADN sin "ATG" o "TAA", ADN con ATG, TAA y la subcadena entre ellos es un múltiplo de 3 (un gen), y el ADN con ATG, TAA y la subcadena entre ellos no es un múltiplo de 3. Para cada cadena de ADN debe:

Imprime la cadena de ADN.

Vea si hay un gen llamando a findSimpleGene con esta cadena como parámetro. Si existe un gen siguiendo nuestro algoritmo anterior, imprima el gen, de lo contrario imprima la cadena vacía.

Parte 2: Encontrar un gen - Usando el algoritmo simplificado reorganizado

Esta asignación determinará si una cadena de ADN tiene un gen utilizando el algoritmo simplificado de la lección, pero organizando el código de una manera ligeramente diferente. Modificará el método findSimpleGene para tener tres parámetros, uno para la cadena de ADN, uno para el codón de inicio y otro para el codón de parada.

Específicamente, debe hacer lo siguiente:

1. Cree una nueva clase Java llamada Part2 en el proyecto StringsFirstAssignments.

2. Copie y pegue los dos métodos findSimpleGene y testSimpleGene de la clase Part1 en la clase Part2.

3. El método findSimpleGene tiene un parámetro para la cadena de ADN llamada ADN. Modifique findSimpleGene para agregar dos parámetros adicionales, uno llamado startCodon para el codón de inicio y otro llamado stopCodon para el codón de parada. ¿Qué cambios adicionales necesita hacer para que el programa se compile? Después de hacer todos los cambios, ejecute su programa para verificar que obtenga el mismo resultado que antes.

4. Modifique el método findSimpleGene para trabajar con cadenas de ADN que sean letras mayúsculas como "ATGGGTTAAGTC" o todas las letras minúsculas como "gatgctataat". Llamar a findSimpleGene con "ATGGGTTAAGTC" debería devolver la respuesta con letras mayúsculas, el gen "ATGGGTTAA", y llamar a findSimpleGene con "gatgctataat" debería devolver la respuesta con letras minúsculas, el gen "atgctataa". SUGERENCIA: hay dos métodos de cadena para UpperCase () y toLowerCase (). Si dna es la cadena "ATGTAA", entonces dna.toLowerCase () da como resultado la cadena "atgtaa".

Parte 3: resolución de problemas con cadenas

Esta asignación le dará práctica adicional usando métodos de cadena. Escribirás dos métodos para resolver algunos problemas usando cadenas y un tercer método para probar estos dos métodos.

Específicamente, debe hacer lo siguiente:

1. Cree una nueva clase Java llamada Part3 en el proyecto StringsFirstAssignments. Ponga los siguientes métodos en esta clase.

2. Escriba el método llamado twoOccurrences que tiene dos parámetros de cadena llamados stringa y stringb. Este método devuelve verdadero si stringa aparece al menos dos veces en stringb; de lo contrario, devuelve falso. Por ejemplo, la llamada twoOccurrences (“by”, “A story by Abby Long”) devuelve verdadero ya que hay dos ocurrencias de “by”, la llamada twoOccurrences (“a”, “banana”) devuelve verdadero ya que hay tres ocurrencias de "a" para que "a" ocurra al menos dos veces, y la llamada twoOccurrences ("atg", "ctgtatgta") devuelve falso ya que solo hay una ocurrencia de "atg".

3. Escriba el método vacío denominado prueba que no tiene parámetros. Este método debe llamar a dosOcurrencias en varios pares de cadenas e imprimir las cadenas y el resultado de llamar a dosOcurrencias (verdadero o falso) para cada par. Asegúrese de probar ejemplos que deberían dar como resultado verdadero y ejemplos que deberían dar como resultado falso.

4. Escriba el método llamado lastPart que tiene dos parámetros de cadena llamados stringa y stringb. Este método encuentra la primera aparición de stringa en stringb, y devuelve la parte de stringb que sigue a stringa. Si stringa no aparece en stringb, entonces devuelve stringb. Por ejemplo, la llamada lastPart ("an", "banana") devuelve la cadena "ana", la parte de la cadena después de la primera "an". La llamada lastPart ("zoo", "forest") devuelve la cadena "forest" ya que "zoo" no aparece en esa palabra.

5. Agregue código al método de prueba para llamar al método lastPart con varios pares de cadenas. Para cada llamada imprima las cadenas pasadas y el resultado. Por ejemplo, la salida para las dos llamadas anteriores podría ser:

La parte de la cuerda después de un plátano es ana.

La parte de la cadena después del zoológico en el bosque es el bosque.

Parte 4: Encontrar enlaces web

Escriba un programa que lea las líneas del archivo en esta ubicación URL, http://www.dukelearntoprogram.com/course2/data/manylinks.html, e imprima cada URL en la página que es un enlace a youtube.com. Suponga que un enlace a youtube.com no tiene espacios y estaría en el formato (donde [cosas] representa caracteres que no son textuales): "http: [cosas] youtube.com [cosas]"

Aquí hay sugerencias para comenzar.

1. Cree una nueva clase Java llamada Part4 en el proyecto StringsFirstAssignments y coloque su código en esa clase.

2. Utilice URLResource para leer el archivo en http://www.dukelearntoprogram.com/course2/data/manylinks.html palabra por palabra.

3. Para cada palabra, verifique si "youtube.com" está en ella. Si es así, busque la comilla doble a la izquierda y derecha de la aparición de "youtube.com" para identificar el principio y el final de la URL. Tenga en cuenta que la comilla doble es un carácter especial en Java. Para buscar una comilla doble, busque (\ "), ya que el carácter de barra diagonal inversa (\) indica que queremos las comillas literales (") y no el carácter Java. La cadena que busca se escribiría "\" "para una comilla doble.

4. Además del método String indexOf (x, num), puede considerar usar el método String lastIndexOf (s, num) que se puede usar con dos parámetros sy num. El parámetro s es la cadena o el carácter a buscar, y num es la última posición en la cadena a buscar. Este método devuelve la última coincidencia desde el inicio de la cadena hasta la posición numérica, por lo que es una buena opción para encontrar la comilla de apertura de una cadena que busca hacia atrás desde "youtube.com". Puede encontrar más información sobre los métodos de String en la documentación de Java para Strings: http://docs.oracle.com/javase/7/docs/api/java/lang/String.html.

Precaución: la palabra Youtube puede aparecer en diferentes casos, como YouTube, youtube o YOUTUBE. Puede encontrar las URL más fácilmente convirtiendo la cadena a minúsculas. Sin embargo, necesitará la cadena original (con letras mayúsculas y minúsculas) para ver la URL de YouTube para responder una pregunta de prueba porque los enlaces de YouTube distinguen entre mayúsculas y minúsculas. El enlace https://www.youtube.com/watch?v=ji5\_MqicxSo es diferente al enlace https://www.youtube.com/watch?v=ji5\_mqicxso, donde todas las letras están en minúsculas.