**Sesión 6: CT aplicado a Humanidades (Digital): Strings y Characters, Detectar letras dobles.**

Un String es una secuencia de caracteres, indicada con una cita (Comillas), ya sean simples o dobles, siempre y cuando las comillas iniciales y finales sean las mismas. Algunos ejemplos de Strings son `hello world’ y `2112'. Una String no puede tener en ella-"-y en ese caso se llama la String nula.

Las Strings también se pueden asignar a variables, por supuesto:

Las Strings pueden ser miles, o decenas de miles, de caracteres de largo, o incluso más largos. Un todo el archivo se puede leer fácilmente en una String; esto se hace comúnmente, y lo haremos más tarde como se describe a continuación analizamos y manipulamos textos más grandes.

En Python, también podemos acceder fácilmente a un carácter de una String utilizando su posición de índice en la String

El primer carácter de una String es el número cero: el índice de la misma es 0. Cuando usted está

contando limas en un mercado, probablemente no empieces a contar "cero, uno, dos..." Pero

hay razones de peso para empezar de cero cuando se trata de los caracteres de una String.

El número de carácter cero es el carácter que está a cero del comienzo de la String.

El índice, en este caso, indica hasta dónde debe llegar (empezando por el principio de

la String) para llegar al carácter en cuestión. Esto se puede considerar como una compensación. ¿Hasta dónde

para llegar al primer personaje? ¡Cero pasos! Ya estás allí. ¿A qué distancia se llega para llegar a la

segundo personaje? Un paso te lleva allí. Así que ese segundo personaje, a un lugar de distancia de

el principio, es el carácter uno, con el índice 1

Selección de una rebanada

Para elegir un personaje, es fácil usar el índice. Podemos conseguir cualquier número de

de los personajes tomando una rebanada. Por ejemplo, los tres primeros caracteres de un string corresponden a

a la rebanada que comienza en 0 y termina en (subiendo a, pero no incluyendo) 3:

Es posible omitir el primer o último número cuando se corta una String, si se desea

indican "desde el principio" o "hasta el final". Para encontrar los tres primeros caracteres, para

también se puede escribir a máquina:

Y para encontrar todo excepto el primer personaje:

Detección de letras dobles

Antes de seguir adelante, podemos poner en práctica nuestros conocimientos sobre las lonchas. Específicamente, podemos escribir un que nos dice cuántas letras dobles hay en una String.

Aunque no importa para esta entrada específica, en general, queremos considerar mayúsculas y minúsculas para que sean iguales, de modo que si nuestra String comienza con "Oolong" nosotros identificará correctamente los dos primeros caracteres como una letra doble. Para lograr esto, sólo hay que usa el método que devuelve una versión en todas las minúsculas de una String, lower():

Al escribir el código de arriba y observar el resultado, usted debe ver que lower() no cambia el valor de la String original; devuelve una nueva String en minúsculas. Si queremos, sin embargo, podemos sobreescribir el valor original con la String en minúsculas.

Comencemos nuestro proceso de detección de doble letra con una versión aún más simple de la más bien función simple que estamos tratando de escribir. Vamos a iterar a través de Wyatt y simplemente encontrar todos los ocurrencias de una doble 'e'. Para ello, nuestro programa debe pasar por el carácter de String por carácter, y en cada punto, debe considerar los dos caracteres siguientes y comprobar que si la String que las contiene es 'ee'. Para empezar, ni siquiera escribiremos una función, aunque lo haremos en algún momento. Por ahora sólo codificaremos Wyatt en un bucle.

La declaración de impresión nos muestra qué valor está contenido en c en cada paso. Aquí hay un código que hace lo mismo: escríbelo y ejecútalo para confirmarlo:

Modifiquemos esto ligeramente para ver qué está pasando:

Tenemos i, un entero, antes de una coma y wyatt[i], un carácter, después de ella. Puede imprimir

cualquier secuencia de valores separándolos con comas, lo que hace que un espacio sea impresa entre ellos. El método de construir cuerdas con algo como str(i) + ': ': '. + wyatt[i] es mejor en muchos casos y es más flexible, pero para echar un vistazo rápido a lo que la valores, esto funciona bien.

Una vez ejecutado esto, se puede ver más claramente que la variable extra, i, que ha sido introducido en este código modificado, se está utilizando como índice. Comienza en 0 y es incrementado hasta llegar a 42, porque esa secuencia es la que se produce por el rango operador:

Así que el rango (len(wyatt)) cuenta los 43 caracteres, comenzando en el primero en el offset 0 y que termina en el último en el offset 42. wyatt[i] entonces recupera el carácter apropiado de la cuerda.

Nuestro primer y sencillo ejemplo sólo podía acceder a un personaje a la vez, y somos interesado en comprobar dos caracteres a la vez. La complejidad que agregamos nos permite hacer que; ahora es simple tomar rebanadas de dos caracteres de la String en lugar de simplemente examinar un carácter a la vez.

El primer problema (contar la doble e) está casi resuelto; ahora podemos identificar cada uno de ellos un par de letras. Sólo necesitamos que el programa cuente cuántos de ellos son'ee'. Vamos a añadir un contador para hacer un seguimiento de lo que encontremos, y añadiremos uno cada vez que encontremos un doble e.

Ahora tenemos código (aún no incluido en una función) que contará cuántas veces hay dos "e" consecutivas. Pero estamos buscando contar todas las letras dobles, incluyendo la letra doble o en "pie". Necesitamos reemplazar la condición wyatt[i:i+2] == 'ee' por una condición apropiada. Lo que realmente queremos probar es si el i-ésimo carácter, el carácter número i, es el mismo que el número del carácter i+1. Intentemos la forma más directa de probando eso:

Interesante.... el par tiene el valor correcto, pero este código resultó en un "índice de Strings de texto de error "range". Por lo tanto, el programa debe haber intentado acceder a un personaje que no existe. Desde que parece haber hecho su trabajo primero, puede haber intentado acceder a un personaje que está más allá el último carácter de la String. Recuerden que empiezo en 0 y subo a 42, la final. En la String es el número 42. Pero nuestro programa no sólo se refiere a wyatt[i], también se refiere al vatio[i+1]. Así que en el último paso, está intentando acceder al carácter 42 y carácter 43 del ejemplo. Sin embargo, no hay ningún carácter 43. Eso es lo que está causando que el error.

Hay algunas maneras de arreglar el problema con la comparación final. Esencialmente, el no debería intentar hacer esta comparación. Es preguntar "¿es el final de esta cuerda? el mismo carácter que el carácter después de él?" Esa pregunta no tiene sentido. Todos los demás las comparaciones tienen sentido, pero no esta última. Así que, hagamos una comparación menos.

En lugar de tener len(wyatt) determinar el límite de nuestra iteración, usaremos len(wyatt)-1.