* String data type
  + A **string** is a sequence of characters
    - Can be letters, numbers, or special characters (punctuation marks and the like)
  + String literals use either single or double quotes
  + + means concatenate for strings
  + Even if a string contains numbers, it’s still considered a string
  + Strings containing numbers (but no letters or special characters) can be converted into an int or float using int() or float()
* Reading and converting strings
  + Data is generally input as a string and then parsed and converted as needed
  + Input numbers have to be converted from strings using float() or int()
* Looking inside strings
  + You can find a single character in a string using an index specified in square brackets
    - fruit = ‘banana’  
      letter = fruit[1]
  + The index value must be a value and starts at zero
    - b is 0, a is 1, n is 2, a is 3, n is 4, a is 5
  + Index value can be an expression that’s computed
    - x = 3  
      w = fruit[x-1]
  + Index value is usually pronounced sub
    - fruit sub 1
  + If you attempt to index beyond end of string you’ll get an error
* String functions
  + **len()** gives the length of a string
* Looking deeper into **in**
  + for letter in ‘banana’ :  
     print(letter)
  + Iteration variable iterates through the sequence
  + The block (body) of code is executed once for each value in the sequence
  + The iteration variable moves through all of the values in the sequence
* Slicing strings
  + We can also look at any continuous section of a string using a colon operator
  + The second number is one beyond the end of the slice (up to but not including)
    - 0:4 would return 0, 1, 2, and 3
  + If second number is beyond end of the string, it stops at the end
  + If we leave off the first or last number, it’s assumed to be the beginning or end of the string
    - [:2] would return 0 and 1
    - [8:] would return from 8 to the end
    - [:] would return the entire string
* Concatenation
  + Strings are **concatenated** using the + operator
    - a = ‘Hello’  
      b = a + ‘There’
  + If we want a space between the two strings when they’re concatenated we have to explicitly insert it
    - a = ‘Hello’  
      b = a + ‘ ‘ + ‘There’
* In as a logical operator
  + The **in** keyword can be used to see if one string is in another
  + This is a logical expression that returns True or False and can be used in an if statement
  + We can also use **not in** to check for the absence of a string
* String comparisons
  + We can use == to see if two strings are the same
  + We can use < or > to see if the string comes before or after
    - This is based on max() and min()
    - Upper case letters are usually less than lower case
      * A < Z
      * a < z
      * A < a
      * Z < a
* String Library
  + Python has a number of string functions in the string library
  + These functions are already built into every string
    - They’re called by appending the function to the string variable
  + These functions don’t modify the original string, they just return a new string that’s been altered
  + The **dir()** function returns a list of the methods available for that class type (string, int, float, etc.)
    - dir() takes either a literal or variable as an argument
* Searching a string
  + The **find()** function can be used to search for a substring within another string
  + **find()** find the first occurrence of the substring
    - If the substring isn’t found, find() returns -1 since string position starts at 0
* Search and replace
  + **replace()** acts like a “search and replace” operation in a word processor
  + It replaces *all* occurrences of the search string with the replacement string
  + The first argument is the substring we want to remove, the second argument is what we want in its place
* Removing whitespace
  + Sometimes we want to take a string and remove whitespace at the beginning of end
  + **lstrip()** and **rstrip()** remove whitespace from the left or right, respectively
  + Whitespace is primarily spaces and tabs, but can include other characters
* Prefixes
  + Checks if a string starts with a substring
    - Capitalization matters!
  + For strVar = ‘Hello world’
    - strVar.startswith(‘Hello’) would be true
    - strVar.startswith(‘h’) would be false
* Parsing and extracting