Strings by themselves are not too interesting, but by processing with strings we do useful things!

- form letters with mail-merge
- advanced processing of spreadsheet data (.csv files)
- text-based games, e.g., hangman

When processing strings, we often times need *data structures* to store a collection of them: list or dictionary.
In the lecture 24 lab for example, the extra credit `sentenceToSpeech.py` program:

- takes in a sentence as a string: `"Mark Guzdial made me President"`
- splits the sentence up into a list of words: `wordList: ['mark', 'guzdial', 'made', 'me', 'president']`
- for each word in the list:
  - concatenate `".wav"`: `'mark.wav'`
  - check to see if that sound file exists. If it does, splice the word's sound to the end of the sentence; otherwise print a message about file not existing.
- play the sentence sound

```
import os
import os.path

""" Splice single word sounds from a sentence together to form a single sentence sound """
def main():
  print "Select the Media Folder"
  setMediaFolder()
  selectedFolder = getMediaFolder()
  print "selectedMediafolder",selectedFolder

  os.chdir(selectedFolder)
  print "cwd", os.getcwd()
  sentenceString = requestString("Enter sentence to say")
  wordList = sentenceString.lower().split()
  print "wordList:",wordList

  sentenceSound = makeEmptySound(1)
  for word in wordList:
    fileName = word + ".wav"
    if os.path.exists(fileName):
      wordSound = makeSound(fileName)
      normalize(wordSound)
      sentenceSound = splice(sentenceSound, wordSound)
    else:
      print 'Sorry the .wav file for the word "'+word+'" could not be found.'

  blockingPlay(sentenceSound)
```

1. At the `requestString` assume the user entered: `"Mark Guzdial made me President"`

For the assignment statement: `wordList = sentenceString.lower().split()`

a) What is the order of operations/method calls?

b) What is the purpose of each method call?

2. How could we splice the words to the sentence in reverse order?

3. On my office computer the `print` statements output:

```
selectedMediafolder C:\Users\fienup\Desktop\Data_Courses\cs1120\lecture_s15\lec24\
cwd C:\Users\fienup\Desktop\Data_Courses\cs1120\lecture_s15\lec24
```

a) How does this relate to the file structure on my computer?

b) Which folder do JES functions (e.g, `makeSound(fileName)`) look for the specified `fileName`?

c) Which folder do `os.path` functions (e.g, `os.path.exists(fileName)`) look for the specified `fileName`?

d) In the program what do you suppose is the purpose of the line: `os.chdir(selectedFolder)`

Below is a summary of the important file-system functions from the `os` module in Python.

`os` Module File-system Functions	
General syntax	**Description**
`getcwd( )`	Returns the complete path of the current working directory
`chdir(path)`	Changes the current working directory to path
`listdir(path)`	Returns a list of the names in directory named path
`mkdir(path)`	Creates a new directory named path and places it in the current working directory
`rmdir(path)`	Removes the directory named path from the current working directory
`remove(path)`	Removes the file named path from the current working directory
`rename(old, new)`	Renames the file or directory named old to new

`os.path` Module File-system Functions	
General syntax	**Description**
`exists(path)`	Returns True if path exists and False otherwise
`isdir(path)`	Returns True if path is a directory and False otherwise
`isfile(path)`	Returns True if path is a file and False otherwise
`getsize(path)`	Returns the size in bytes of the object named path

Note: On most operating systems, `"."` represents the current folder and `".."` represents its parent folder.

4. If the "current working directory (cwd) is:

`C:\Users\fienup\Desktop\Data_Courses\cs1120\lecture_s15\lec24`

which directory would be the current working directory after the statements:

```
os.chdir("..\\..")
os.chdir(r"..\..")
```