While loops
Files

Concep Question

```
def hasVowels(word):
    if type(word) == str:
        for letter in word:
            if letter in 'aeiou':
                return True
    else:
        return False
```

What is the return value for hasVowels("")?

A. True

B. False

C. None

Motivating While Loops

- So far, we know about one type of loop: `for` loop
  - It requires a sequence (e.g. a `range` sequence or a string) to loop over
- Another type of loop is the `while` loop: it repeatedly tests a condition, executing the entire body of the loop if it is True, and terminating the loop if it is False
  - Useful when there is no sequence to loop over
  - Commonly used when we don't know how many times the loop will run

ConcepTest

What is printed by the following code? (Output is on one line to save space.)

```
x = 6
while x > 4:
  print(x)
  x = x - 1
```

- A. 6 5
- B. 6 5 4
- C. 5 4
- D. 5 4 3
- E. 6 5 4 3

ConcepTest

What is printed by the following code? (Output is on one line to save space.)

```
x = 6
while x > 4:
  x = x - 1
  print(x)
```

- A. 6 5
- B. 6 5 4
- C. 5 4
- D. 5 4 3
- E. 6 5 4 3

For vs. While

Use `for` when:

- You want to loop through an entire sequence without stopping
- The number of iterations does not depend on user input
- The increment to the loop variable is the same on every iteration

```
s = 'abc'
for count in range(len(s)):
  print('Index {0} is {1}'.format(count, s[count]))

count = 0
while count < len(s):
  print('Index {0} is {1}'.format(count, s[count]))
  count += 1
```

ConcepTest

```
valid = False
while not valid:
  s = input ("Enter a password: ")
  valid = len(s) == 5 and s[:2] == 'xy'
```

Which of the following passwords gets us out of the loop?

- A. `xyz`
- B. `abcxy`
- C. `xyabc`
- D. More than one of the above passwords get us out of the loop
- E. None; the loop never executes and no passwords are obtained

True and break

- There are several ways to write a loop whose body is required to run at least once
  1. Artificially make the condition true before the loop starts (like `inputloop.py`)
  2. Copy some loop code above the loop to make the condition true
  3. Use `True` as the condition and `break` to exit the loop
- `break` causes immediate termination of the loop
- `break` can make code difficult to read if used improperly
- We frequently do not allow `break` on exams or assignments

ConcepTest

A valid password is one that is length 5 and starts with `xy`. Such passwords should get us out of the loop. Which of these does this?

- A.

```
while True:
  s = input ("Enter a password: ")
  if len(s) == 5 and s[:2] == 'xy':
    break
```

- B.

```
s = input ("Enter a password: ")
while len(s) == 5 and s[:2] == 'xy':
  s = input ("Enter a password: ")
```

- C. Both are correct
- D. None is correct

ConcepTest

What is the output of this code? (Output is on one line here to save space.)

```
n = 3
while n > 0:
  if n == 5:
    n = -99
  print(n)
  n = n + 1
```

- A. 3 4
- B. 3 4 5
- C. 3 4 -99
- D. 3 4 5 -99

Files

- Files give us PERSISTENCE
  - Data in programs is cleared with every run, not the case with files
- Text files provide convenient input/output storage
  - e.g. programs can read configuration data or input files to process, and can write output to files

Files – important terms

- File: A document
- Directory: A folder containing files and other folders
- File System: Collection of all the files and folders on the computer, organized in a hierarchy

File Input/Output

- We read data from a file into our program.
- We write data from our program into a file.
- Steps for File I/O

  1. Open the file (creates a "connection" between your program and the file).

```
f = open('animals.txt')
```

  2. Read the data / write the data

  3. Close the file (close the "connection"). This should to be done once per file.

Reading Files with Methods

- Several methods for reading text from files:
  - `readline()`: reads and returns next line; returns empty string at end-of-file
  - `read()`: reads the entire file into one string
  - `readlines()`: reads the entire file into a list of strings
- All of these leave a trailing '\n' character at the end of each line.

```
f = open('animals.txt')
line = f.readline()
print(line)
line = f.readline()
f.close()
```

Reading Files in a loop

```
f = open('animals.txt')
for line in f:
     print(line.strip())
f.close()
```

See detailed lecture notes for usage with read and readlines

Writing to file

```
outfile = open('example_2.txt', 'w')
outfile.write("Duck\nCow\nCat")
outfile.close()
```

HW Question

a. (10 pts) The first way is shown in the listing at the bottom of p. 112. On line 4 of that listing we see:

```
content = infile.read()
```

After this line of code is executed, what would `type(content)` return? (i.e. would it be `<class 'int'>`, `<class 'float'>`, `<class 'str'>`, `<class 'list'>`, or something else?)

b. (10 pts) The second way is shown in the middle of p. 113. On line 7 of that listing we see:

```
wordList = content.split()
```

What does the `.split` method do, and what is stored in `wordList` as a result?

HW Questions

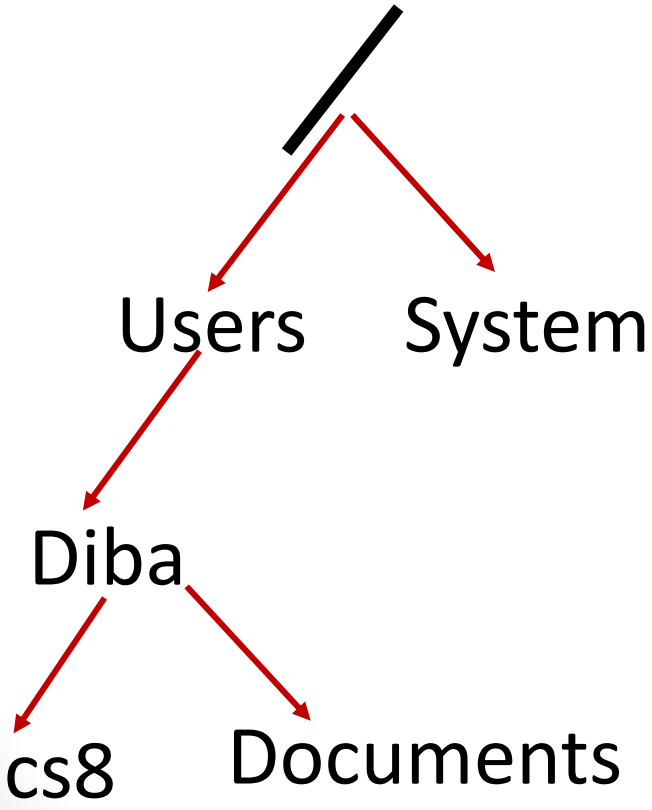
d. (10 pts) The fourth and final way is shown in an interactive example on the lower half of p. 114, and looks like this (with the Python prompts removed):

```
infile = open('example.txt')
for line in infile:
   print(line,end='')
```

The book suggests that this fourth method has an advantage over the other three in a particular circumstance. What is the circumstance in which we would want to use this method instead of one of the other three?

Unix File System

- Root (/)
- Path



Concept Question

Every file on a file system can be referred to be an "absolute pathname", which consists of a sequence of … what?

A. Files
B. Directories
C. Paths

Concept Question

In contrast to an “absolute pathname”, we have the concept of a “relative pathname”. What is the technical term used for the “starting point” of a “relative pathname”?

A. Root
B. Home directory
C. Current directory
D. None of the above

Navigating the unix file system

- Some common unix commands
  - `ls`
  - `pwd`
  - `mkdir`
  - `cd`