File IO
String formatting

Loops: Review

```
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

Files

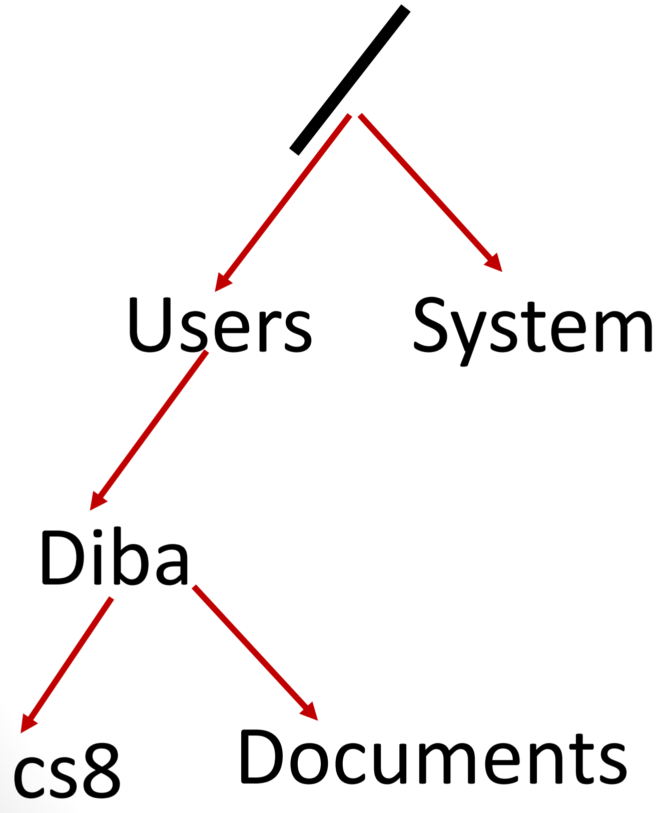
- Files give us PERSISTENCE
  - Data in programs is cleared with every run, not the case with files
- Text files provide convenient input/output storage
  - e.g. programs can read configuration data or input files to process, and can write output to files

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

Unix File System

- Root (/)
- Path



HW07 Q2(b)

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

HW07 Q2(c)

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”?

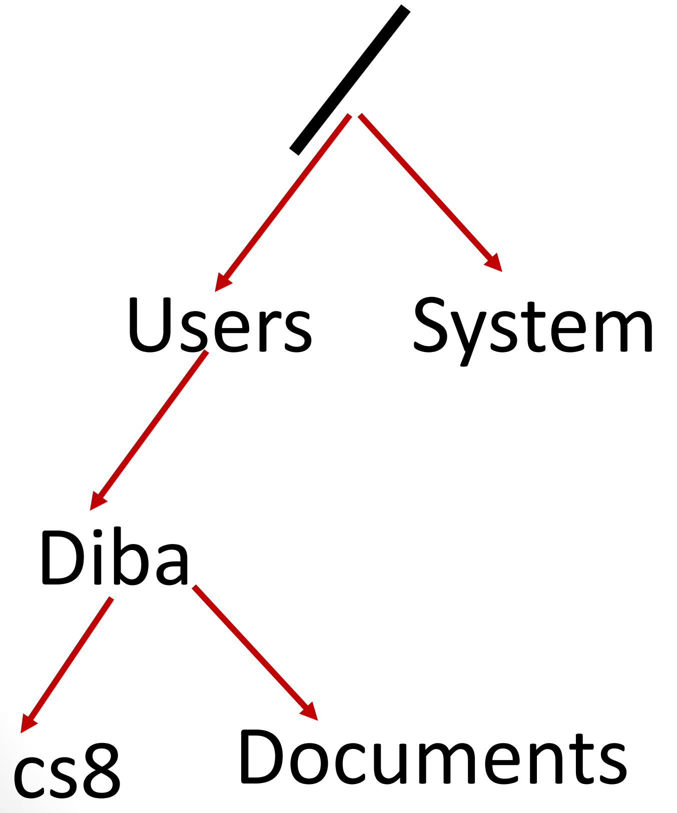
A. Root
B. Home directory
C. Current directory
D. None of the above

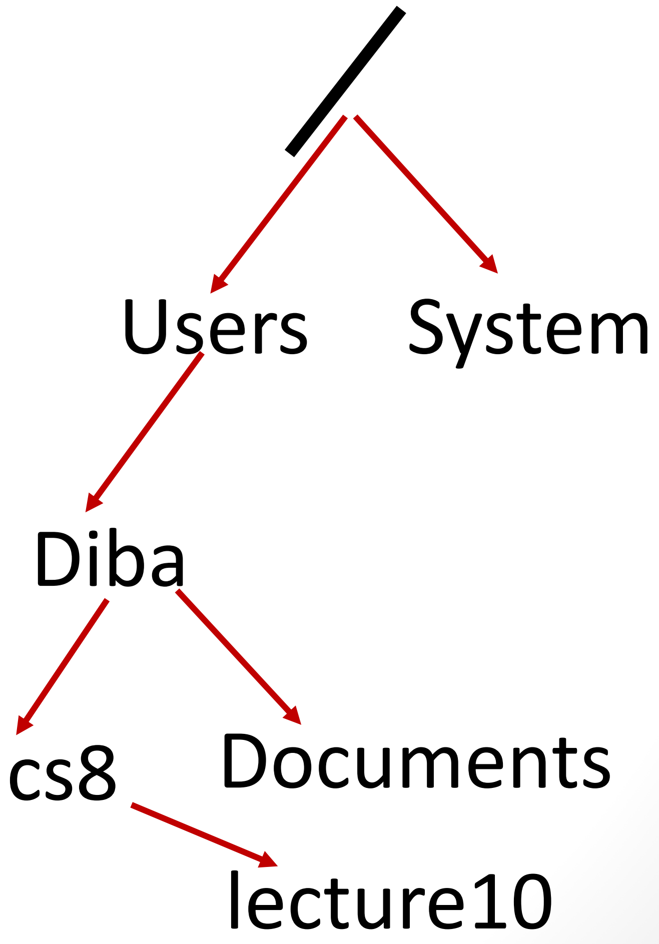
Navigating the unix file system

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

Concept Question

Write the unix commands converts the file system on the left to the one on the right (assume you are in /Users)?





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

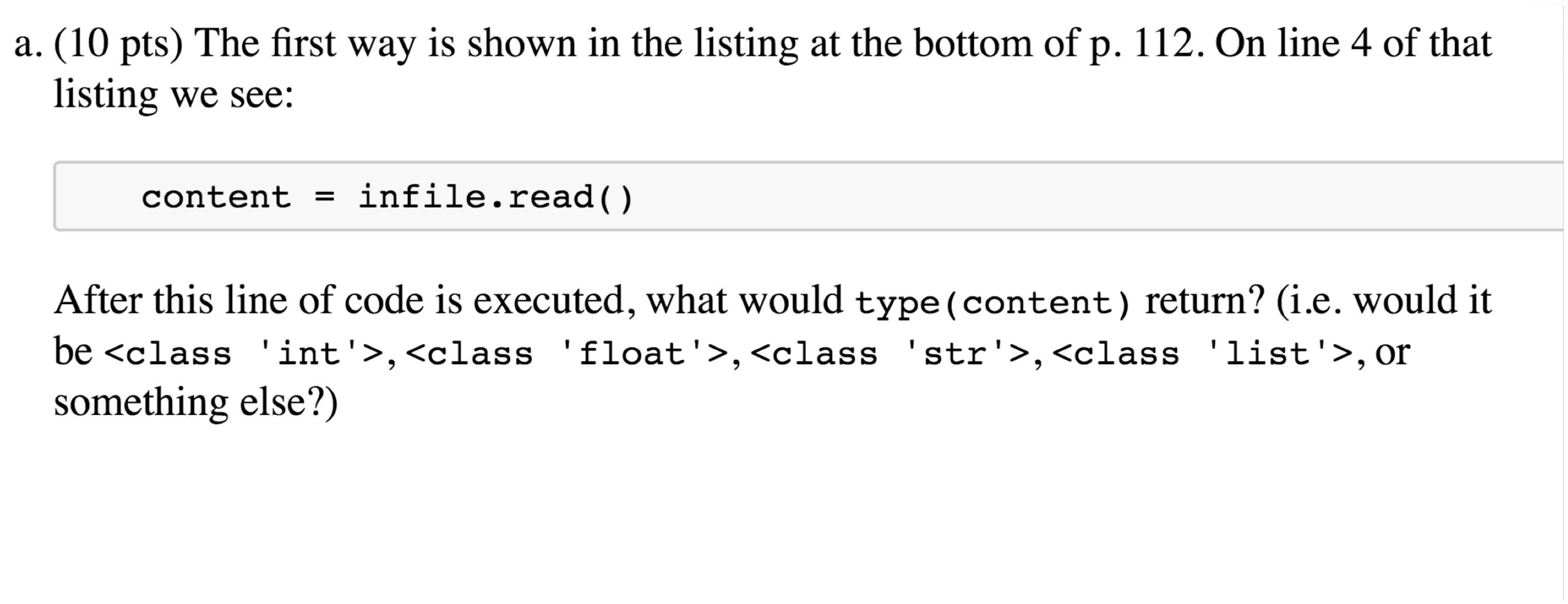
```
f = open(‘animals.txt’)
for line in f:
    print(line.strip())
f.close()
```

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()
```

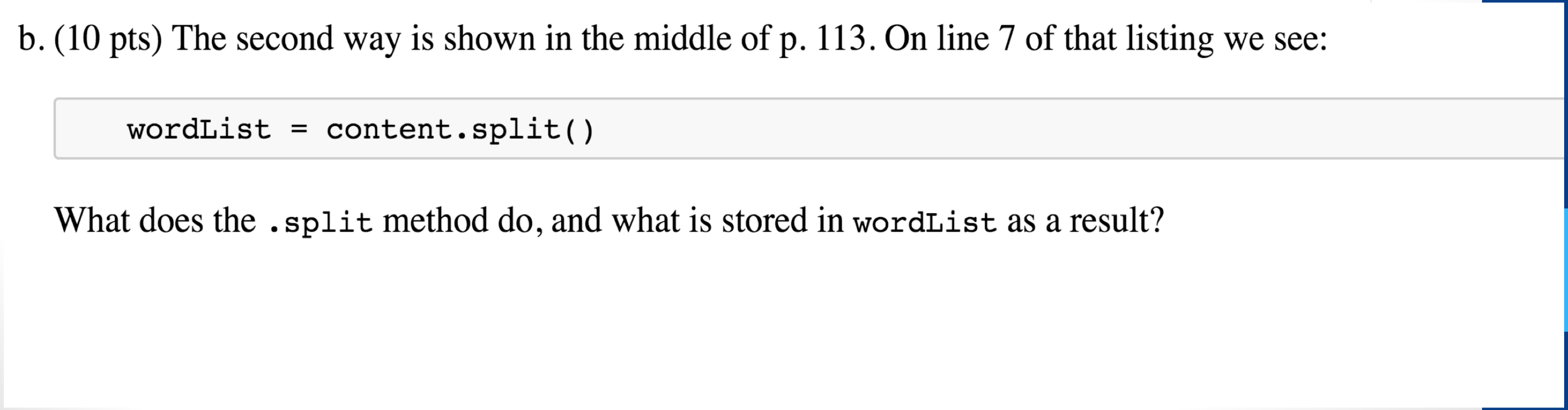
HW07



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?

HW07

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?

String Methods

```
s = "CS 8: Intro to Programming”
s.find(“8”)
s.find(“Math”)
s.startswith(“CS”)
s.startswith(”Computer”)
s.endswith(“ing”)
s.endswith(“Prog”)
s.count(‘m’)
'Mississippi'.count('i’)
s.replace(":", "#")
s.upper()
‘Mississippi’.lower()
```

Concept Question

```
MS = "Mississippi"
MS.replace("i", "!")
print(MS)
```

What is printed?

A. Mississippi

B. M!ss!ss!pp!

C. Error

D. None of the above

String formatting

Let’s say you have an integer price:

```
price = 18.00
```

Write a statement to print:

```
The price is <price>. Wow that’s cheap!
```

''' Format specification:

{ : }. Left side of colon say which argument to place into {}

To the right we specify a FIELD WIDTH (i.e., how many spaces/ columns on the screen to devote to this

```
print("-->{}<--".format(price))
print("-->{:20}<--".format(price))
```

HW07

3. (10 pts) Section 4.2 discusses formatted output. What is the output of the following print statement? Please put one character per box to show the exact spacing. Try to figure it out by hand before checking your answer online. If you spoil the first grid, use the second.

```
print('{0:4},{2:6}'.format(123,456,789))
```