CS 1120: Media Computation Spring 2018
Lab 11: Introduction to strings

There is a lot of good material in chapter 3 of your textbook. This lab is going to focus on the parts that are the most essential for you to have a basic understanding of how text can be manipulated programmatically.

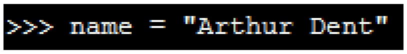
Let’s begin with a basic definition from your textbook:

"Text is typically manipulated as strings. A string is a sequence of characters. Strings are stored in memory as an array, just like as sounds."

Just about everything that you need to know about working with strings and text files is contained in those three sentences.

First of all, as you observed at the beginning of the course and now again in this chapter, a string is a sequence of characters defined inside of quotation marks. In Python, we can use either single quotes (the apostrophe), double quotes (what you normally think of with quotation marks, or triple quotes (actually three single or double quotes in a row). I would encourage you to use double quotes in this lab: these are valid in nearly every programming language you will work with.

So for example, I might define a variable to hold someone's name by writing:



```
>>> name = "Arthur Dent"
```

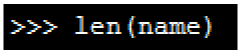
This then defines a string called name which consists of a sequence of characters. Once we have this we have a data structure very similar to what we had when we were working with pictures and sounds. In fact, we can look at a string in similar manners.

One of the first things we might want to do with a string is ask about its size. Think about this – when we work with pictures and sounds we needed to know their size. This is a good review.

[Q1] What command tells us the size (length) of a sound file?

[Q2] What TWO commands told us the size (dimensions) of a picture file?

We can see the comparable information for a string by asking the computer the following:



```
>>> len(name)
```

[Q3] What answer came back?

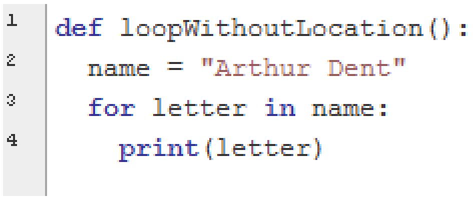
[Q4] Why does this answer make sense?

One of the next things you learned how to do when working with picture and sound files was to write a loop that would work through all of the data in the picture/sound. With both of those there were two ways to do this:

[Q5] What were the two ways that you could work through the pixels in a picture? Write a little sample code for each technique.

[Q6] What were the two ways that you could work through the sound samples in a sound? Write a little sample code for each technique.

It turns out there are also two different ways to work your way through the information in a string. Furthermore, they are the same overall strategies you should have answered in Q5 and Q6. First, you can do it without worrying about the specific location of the data. To demonstrate this, type the following program into the Program Area of JES and save this as lab11.py.

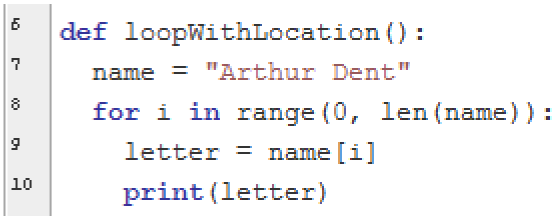


```
def loopWithoutLocation():
  name = "Arthur Dent"
  for letter in name:
    print(letter)
```

Run this program.

[Q7] What do you observe when you run this program?

The second way to work through the data in a string is to set up a loop that counts through the indeces of the string one number at a time and understands the location of each character in the string. Add the following program to lab11.py



```
def loopWithLocation():
  name = "Arthur Dent"
  for i in range(0, len(name)):
    letter = name[i]
    print(letter)
```

[Q8] What do you observe when you run this program?

You should notice real similarities to these two programs and the kinds of loops you wrote for Q5 and Q6. You should also notice real similarities with how indexes (variable "i" in our program) work in strings when compared to pictures and sounds. Notice that the FIRST character is actually stored at index 0 and the last character is stored at length-1. This is exactly what is done with sounds and pictures.

Next, let’s look at how you look at one small part of a string. As we have been doing, it helps if we start by reviewing what we did with pictures and sounds.

[Q9] What function gives you access to one specific pixel in a picture?

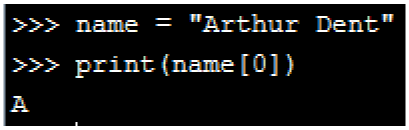
[Q10] Write the function call that would give access to the fourth pixel in a picture called “pic1”

[Q11] What function gives you access to one specific sound sample in a sound?

[Q12] Write the function call that would give access to the fourth sound sample in a sound called “s1”

If you looked closely at loopWithLocation() when you typed it into JES you may have noticed that we access characters in a string in a much more direct manner. We DO NOT do this by using a function. Instead, we simply type up the name of the variable, put a set of square brackets after it, and put the index number inside of the square brackets.

For example, to access the first character in a string called “name” we would type:



```
>>> name = "Arthur Dent"
>>> print(name[0])
A
```

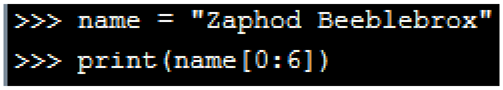
Given that:

[Q13] Write the command that would give access to the fourth character in a string called “name”

[Q14] Write the command that would give access to the LAST character in the string called name as defined above.

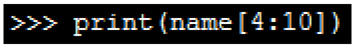
[Q15] Write the command that would give access to the last character in a string called name where you weren’t sure how long it was. (HINT: You can ask the computer how long a string is. We did it in Q3).

Python not only lets you get access to individual characters, but it lets you get access to substrings (pieces of the string you are working with). Type in the following:



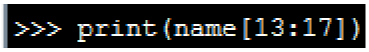
```
>>> name = "Zaphod Beeblebrox"
>>> print(name[0:6])
```

[Q16] What was printed?



```
>>> print(name[4:10])
```

[Q17] What was printed?



```
>>> print(name[13:17])
```

[Q18] What was printed?

[Q19] If I asked for name[a:b] where a and b are numbers, what will be printed?

[Q20] What command would I have to give to print "Bee"?

Now let’s break this down a bit further and consider the individual units of a string. As we have done before, it will help if we review the individual units of pictures and sounds.

[Q21] Pictures are made up of small units called what?

[Q22] Those are further divided into three color channels. What are they?

[Q23] Each color channel can take on any number value between _______ and ______. What numbers belong in the blanks?

[Q24] Each color channel uses how much memory?

[Q25] Sounds are made up of small units called what?

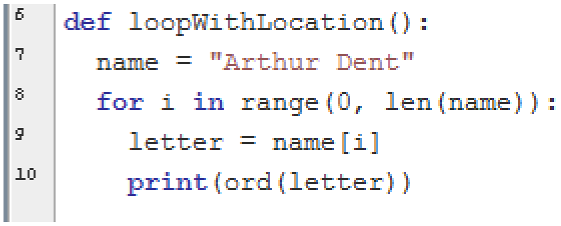
[Q26] **E**ach sound sample can take on any number value between _____ and ____. What numbers belong in the blanks?

[Q27] Each sound sample uses how much memory?

Now let's begin to answer the similar questions for strings.

[Q28] Strings are made up of small units called what? (And no, the answer isn't letters. Why?)

Now at first the next question in the series may seem strange. The next question asks what number values each character can take. But the fact is that each character does have a number value. We can see that with the following modification to your last program:



```
def loopWithLocation():
  name = "Arthur Dent"
  for i in range(0, len(name)):
    letter = name[i]
    print(ord(letter))
```

[Q29] What do you observe when you run this program?

Where do all of those values come from? These are values from the standard encoding for strings known as ASCII. Google both "ASCII" and "ASCII table." Do a little bit of research.

[Q30] What does ASCII stand for?

[Q31] How many characters does ASCII encode?

[Q32] What is the ASCII value for "A"?

[Q33] What is the ASCII value for "Z"?

[Q34] What is the ASCII value for "a"?

[Q35] What is the ASCII value for "z"?

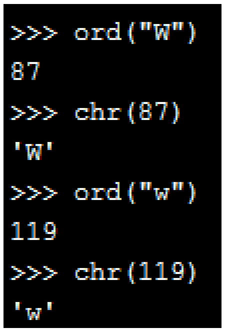
[Q36] What is the ASCII value for the number "0"?

[Q37] What is the ASCII value for "9"?

[Q38] What is the ASCII value for a space?

[Q39] What is the ASCII value for a newline character?

By the way, I should point out that there is a way to reverse this.



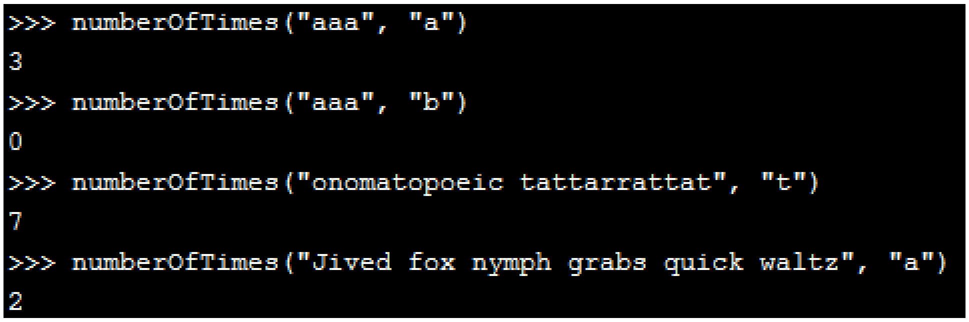
```
>>> ord("W")
87
>>> chr(87)
'W'
>>> ord("w")
119
>>> chr(119)
'w'
```

While ord() tells you the ordinal value of a given character, chr() tells you which character goes with a particular ordinal value.

[Q40] Each character in ASCII can be encoded with a number value between _____ and ____. What numbers belong in the blanks?

[Q41] Each character uses how much memory?

[Q42] Finally, let's write a simple program. In a previous assignment, you counted the number of samples in a sound which had the value of zero. Use the same approach to count the number of times a given character occurrs in a string. Your function should take a string and a character as arguments and print out the number of times that chatracter occurs in the string:



```
>>> numberOfTimes("aaa", "a")
3
>>> numberOfTimes("aaa", "b")
0
>>> numberOfTimes("onomatopoeic tattarrattat", "t")
7
>>> numberOfTimes("Jived fox nymph grabs quick waltz", "a")
2
```

Submit your work

And you're done! You must submit two files to eLearning:
1) Your file lab11.py, with the code you wrote (questions 5, 6 and 42; as well as the sample code you wrote for some questions)
2) Your answers to the questions in this lab using the response word document (you may use as many lines per question as you like).