Activity 1.3.5 Strings

Introduction

Text– it's all around us. Books, web pages, email, text messages, papers for school. These are mostly text, and most of it has been digitized, represented with zeros and ones. Most programming languages have a data type called a string. A string takes care of the details for us, and we can just think of some text as a string of characters. One character might be a letter, number, or symbol, depending on the character set of the representation standard.

STRING CAN KILL.
Please share this photo to help keep cats safe.
If ingested, string, thread, ribbon, yarn, etc can cause life-threatening intestinal issues in cats. Keep your cat safe:
• Never leave string where your cat can get it.
• Never leave your cat alone with toys that have string or yarn.
• If you suspect your cat has swallowed string, call your vet immediately.
• Never pull on a string dangling from your cat's mouth or anus. If you see this, take your cat to the emergency vet immediately.
Following this advice may save your cat's life!
Petfinder

Keep cats off string.

Why do you suppose computer scientists chose the term *string*?

Procedure

1. Form pairs as directed by your teacher. Meet or greet each other to practice professional skills and establish norms.

2. Launch Canopy and open an editor window.

3. Start a new program in the code editor by choosing **File** > **New** > **Python file**. Save the file as `JDoe_BSmith_1_3_5.py`

4. In previous activities you learned that you can make decisions by evaluating Boolean expressions. You also learned input and output.

```
In []:  from __future__ import print_function
In []:  if raw_input('One character: ') == '!':
   ...:     print('Wow', end='!')
One character: !
Wow!
```

5. In addition to the native types we've seen so far (`int`, `float`, `long`, `bool`), another type (`str`) represents strings of characters. You can use the function `type()` to check the variable type of a variable or expression.

```
In []: slogan = 'My school is the best'
In []: type(slogan)
Out[]: str

In []: slogan
Out[]: 'My school is the best'
```

Which of these types can represent the number six million?

6. String literals are enclosed in single or double quotes. The opening and closing quotes must match.

One of the following two inputs will produce an error. Try this, discuss both outputs with your partner, and summarize your discussion.

```
In []: type('tr' + "y this")
In []: type('tr' + 5)
In []: #7. (Discuss and explain.)
```

7. Strings are **iterables**. Iterables are sequences that can be counted in order, one at a time, during iteration. Strings contain a sequence of characters, one after another. The elements — including the spaces — are **indexed**, starting at `0`.

`'My school is the best'`

Character	M	y	space	s	c	h	o	...	t
Index #	0	1	2	3	4	5	6	...	20

Try this, discuss the outputs with your partner, and summarize your discussion.

```
In []: slogan[0]
Out[]: 'M'

In []: slogan[2]

In []: slogan[8]

In []: slogan[26]

In []: slogan[2]

In []: #8. (Discuss and explain.)
```

8. *Python* allows iterables to be **sliced**. To slice, use square brackets and two indices separated by a colon. *Python* returns the iterable from the beginning index up to but not including the ending index.

```
In []: slogan[0:5] # Note that slogan[5] is 'h'
Out[]: 'My sc'

In []: slogan[5:21]
Out[]: 'hool is the best'
```

When slicing, you can omit the starting (or ending) index if you want to start at the beginning (or end at the ending) of the string.

```
In []: slogan[:5]
Out[]: 'My sc'
```

Try to return `'best'` by slicing the variable slogan, omitting the end index.

```
In []: # 9. Slicing
In []: (Write code to return 'best'.)
```

9. Use slicing and **concatenation** to create your own sentence. Concatenation involves pasting together two strings, one after another. Follow the example here.

```
In []: slogan[:13] + 'awesome!'
Out[]: 'My school is awesome!'
```

10. The `len()` function returns the number of elements in an iterable. The index of the last element is always one less than the length of the iterable since the indices begin at 0.

```
In []: len(slogan)
Out[]: 21
```

Explain the output of the following inputs:

```
In []: activity = 'theater'
In []: len(activity)
In []: # 11a. (Discuss and explain.)

In []: activity[0 : len(activity)-1]
In []: # 11b. (Discuss and explain.)
```

11. The `in` keyword can be used as a Boolean condition, returning `True` or `False`:

```
In []: 'test goo' in 'Greatest good for the greatest number!'
Out[]: True
In []: # 12. (Discuss and explain.)
```

12. A social media site offers a contest to write a humorous short paragraph. A constraint on the creative format: the entry must include a question, a quote, a compound sentence, and an exclamation. These would contain the characters `?`, `"`, `,`, and `!`, respectively.

    Create a function `how_eligible(essay)` that returns `0` to `4`, equal to the number of these four characters that the essay included. Generate ideas for how to solve this problem, strategize, and then code and test iteratively.

    ```
    In []: how_eligible('This? "Yes." No, not really!')
    Out[]: 4
    In []: how_eligible('Really, not a compound sentence.')
    Out[]: 1
    ```

Conclusion

1. How many characters are in this sentence? Does it matter whether *Python* is storing the string as one byte per character or four bytes per character?

2. This question asks you about something you have not learned. In fact, the question is asking about details that go beyond what you will learn in this course. However, wondering what is going on at a lower level of abstraction – and talking about it – can be a useful strategy when learning about computing.

   Describe what you think occurs in memory when the following code is executed.

```
In []: a = 'one string'
In []: b = 'another'
In []: c = a[:3] + ' and ' + b
In []: print(c[6:10])
```