Study groups sign-up is open for all courses, including CSCI 203.

https://buapps.bucknell.edu/script/studentlearningsupport/groupstudy/

Data types

Python maintains data by **`type`** :

Numeric types	`float`	`>>> type(3.14)`
		`<class 'float'>`

	`int`	

	`bool`	`>>> type(True)`
		`<class 'bool'>`

Sequence types	`str`	`>>> type('writer')`
		`<class 'str'>`

	`list`	`>>> type([1,2,3])`
		`<class 'list'>`

`str`ing functions

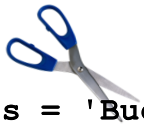
`str`	`str(42)` returns `'42'`	converts input to a string
`len`	`len('42')` returns `2`	returns the string's length
`+`	`'XL' + 'II'` returns `'XLII'`	*concatenates* strings
`*`	`'VI'*7` returns `'VIVIVIVIVIVIVI'`	*repeats* strings

Given these strings { `s1 = "ha"` , `s2 = "t"` }

What are the following strings?

`s1 + s2`	`hat`
`2*s1 + s2 + 2*(s1+s2)`	`hahathathat`

String surgery



```
s = 'Bucknell University'
     0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

`s[ ]` *indexes* into the string, returning a one-character string

index

`s[0]` returns	'B'
`s[6]` returns	'l'
`s[11]` returns	'i'
What returns `'v'`?	`s[12]`
`len(s)` returns	19
`s[len(s)]` returns	ERROR

`python !=` English

Negative indices...

```
      0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
s = 'Bucknell University'
    -19 -18 -17 -16 -15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1
```

Negative indices count ***backwards*** from the end!

`s[-1]` returns	'y'
`s[-10]` returns	'U'
`s[-0]` returns	'B'

Slicing

what if you want a bigger piece of the pie???

```
s = 'Bucknell University'
     0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

`s[ : ]` ***slices*** the string, returning a **substring**

the first index is the first character of the slice

the second index is **ONE AFTER** the last character

`s[0:8]`	returns `'Bucknell'`
`s[9:18]`	returns `'Universit'`
`s[17:]`	returns `'ty'`
`s[:]`	returns `'Bucknell University'`

Slicing

```
s = 'Bucknell University'
     0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

`s[ : ]` *slices* the string, returning a substring

What are these slices?

```
s[15:-1]        'sit'
s[4:7]          'nel'
```

How do you get:

```
'Uni'           s[9:12]
'Universe'      s[9:16] + 'e'
                s[9:16] + s[5]
```

Skip-Slicing

if you don't want your neighbor to get any…

```
s = 'Bucknell University'
     0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
```

`s[ : : ]` ***skip-slices***, returning a subsequence

the third index is the "stride" length   it defaults to 1

```
s[0:10:2]       returns 'Bcnl '
s[17:13:-1]     returns 'tisr'
```

What skip-slice returns `'clnr'`   s[2:15:4]

What does this return? `'m' + s[1::6]`   returns `'mule'`

`s[::-1]`   returns `'ytisrevinU llenkcuB'`

Lists → collections of *any* data

```
L = [ 3.14, True, 'third', 42 ]
```

Commas separate elements.

Square brackets tell python you want a list.

```
L = [ 3.14, [2,40], 'third', 42 ]
```

Lists are more general than strings. Strings are always sequences of characters, whereas lists can contain values of any type.

You can have a list in a list!

List operations

```
L = [ 3.14, [2,40], 'third', 42 ]
```

len(L)	L[0]	L[0:1]
length	indexing	slicing

How could you extract from **L**   `'hi'`

List operations

+	*
concatenation	**multiplication**
Joins two lists	*Repeats list a number of times*

```
>>> P = [ 3.14, [2,40], 'third', 42]
>>> R = ['a','b','c']
>>> P + R
[3.14, [2, 40], 'third', 42, 'a', 'b', 'c']
```

```
>>> lst = [1,2,3]
>>> lst * 3
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

The `in` operator – membership testing for lists and strings

```
>>>'i' in 'alien'                          True
>>> 3*'i' in 'alien'                       False
>>> 'i' in 'team'                          False
>>> 'cs' in 'physics'                      True
>>> 'sleep' not in 'CSCI 203'              True
>>> 42 in [41,42,43]                       True
>>> 42 in [ [42], '42' ]                   False
```

Mutable and immutable sequences

Strings are immutable

Once a string is created, individual elements of string or the string as a whole cannot be changed

```
>>> st = 'ABC'
>>> st[0]
'A'
>>> st[0]='B'
Traceback (most recent call last):
  File "<pyshell#33>", line 1, in <module>
    st[0]='B'
TypeError: 'str' object does not support item assignment
```

Lists are mutable

Individual items or entire slices can be replaced through assignment statements

```
>>> lst = ['A', 'B', 'C']
>>> lst
['A', 'B', 'C']
>>> lst[0] = 'B'
>>> lst
['B', 'B', 'C']
```

Raising and razing lists *Answers*

```
pi = [3,1,4,1,5,9]
L = [ 'pi', "isn't", [4,2] ]
M = 'You need parentheses for chemistry !'
     0   4   8   12  16  20  24  28  32
```

Part 1		Part 2	
What is `len(pi)`	6	What is `L[0]`	'pi'
What is `len(L)`	3	What is `L[0:1]`	['pi']
What is `len(L[1])`	**5**	What is `L[0][1]`	'i'
What is `pi[2:4]`	[4,1]	What slice of `M` is `'try'`	M[31:34]
What slice of `pi` is `[3,1,4]`	pi[0:3]	What is `M[9:15]`	'parent'
What slice of `pi` is `[3,4,5]`	pi[::2]	What is `M[::5]`	**'Yeah cs!'**

What are `pi[0]*(pi[1] + pi[2])` and `pi[0]*(pi[1:2] + pi[2:3])` ?

15 **[1,4,1,4,1,4]**