Slide 8. Variables – a way to refer to objects (just like math class)

Type and press enter:

```
In [148]: word = 'python'
```

```
In [149]: phrase = 'is awesome'
```

Notice that there is no output

```
In [150]: number = 5
```

Functions can act on variables. Type and press enter:

```
In [151]: type(word)
```

```
Out[151]: str
```

```
In [152]: type(number)
```

```
Out[152]: int
```

Depending on the object math or string concatenation can occur. Type and press enter:

```
In [153]: word + phrase
```

```
Out[153]: 'pythonis awesome'
```

```
In [154]: number + number
```

```
Out[154]: 10
```

Slide 10. Your turn

You want to load 30.0 ug of protein for a Western and your protein concentration is 2.0 ug/ul. How many ul will you need?

1. Create the variable protein and assign 30.0 and create the variable concentration and assign 2.0
2. Create the variable answer and assign the arithmetic to this variable – what is the answer? (print the variable)

```
In [155]: protein = 30
```

```
In [156]: concentration = 2
```

```
In [157]: answer = protein/concentration
```

```
In [158]: answer
```

```
Out[158]: 15
```

Given the phrase The answer to life, the universe, and everything = and the number 42, assign each to a variable. Using these variables create the phrase “The answer to life, the universe, and everything = 42” as output

```
In [159]: phrase1 = 'The answer to life, the universe, and everything = '
```

```
In [160]: phrase2 =  '42'
```

```
In [161]: wholephrase = phrase1 + phrase2
```

```
In [162]: wholephrase
```

```
Out[162]: 'The answer to life, the universe, and everything = 42'
```

Slide 11. Indexing

- Indexing by position can be used with strings or lists (lists - another object we will cover later)
- Can access individual pieces of a string with indexing
- Type and press enter:

```
In [163]: word
```

```
Out[163]: 'python'
```

```
In [164]: word[0]
```

```
Out[164]: 'p'
```

```
In [165]: word[5]
```

```
Out[165]: 'n'
```

```
In [166]: word[-1]
```

```
Out[166]: 'n'
```

Slide 13. Indexing

Indexing starts at zero goes 1 past the end. Type and press enter:

```
In [167]: word[0:6]
```

```
Out[167]: 'python'
```

```
In [168]: word[6]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-168-9f3bdf74c108> in <module>()
----> 1 word[6]

IndexError: string index out of range
```

If you leave the start or end position blank, python will assume you want to start at the beginning and end at the end. If you leave the increments blank, python defaults to increments of 1. Type and press enter:

In [169]:
```
word[:6]
```

Out[169]:
```
'python'
```

In [170]:
```
word[0:6:2]
```

Out[170]:
```
'pto'
```

In [171]:
```
word[::-1]
```

Out[171]:
```
'nohtyp'
```

Slide 14. Your Turn

Use indexing to output ‘atcg’ from ‘actgcag’ Remember to assign ‘actgcag’ to a variable first

In [172]:
```
sequence = 'actgcag'
```

In [173]:
```
sequence[0:7:2]
```

Out[173]:
```
'atcg'
```

Type and hit enter:

In [174]:
```
dir(word)
```

Out[174]:
```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getnewargs__',
 '__getslice__',
 '__gt__',
 '__hash__',
 '__init__',
 '__le__',
 '__len__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rmod__',
 '__rmul__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 '_formatter_field_name_split',
 '_formatter_parser',
 'capitalize',
```

```
 'center',
 'count',
 'decode',
 'encode',
 'endswith',
 'expandtabs',
 'find',
 'format',
 'index',
 'isalnum',
 'isalpha',
 'isdigit',
 'islower',
 'isspace',
 'istitle',
 'isupper',
 'join',
 'ljust',
 'lower',
 'lstrip',
 'partition',
 'replace',
 'rfind',
 'rindex',
 'rjust',
 'rpartition',
 'rsplit',
 'rstrip',
 'split',
 'splitlines',
 'startswith',
 'strip',
 'swapcase',
 'title',
 'translate',
 'upper',
 'zfill']
```

- dir function outputs the methods that can be used with a particular object – in this case string object
- python uses the . notation for methods that act on objects; syntax is:
- object.method()
- Synonymous to are functions which also act on objects:
- function(object)
- It is easier to google the syntax for function or method than to bother with dir()

Slide 16. Useful methods for strings

Type and press enter:

```
In [175]: len(word)
```

```
Out[175]: 6
```

```
In [176]: word.split('t')
```

```
Out[176]: ['py', 'hon']
```

```
In [177]: word.find('t')
```

```
Out[177]: 2
```

In [178]:
```
word.count('t')
```

Out[178]: 1

Another way to put numbers within strings, type and press enter:

'The answer to life the universe and everything is {0} '.format(6*7)

To exit python, type and press enter:

quit()

Slide 17. Building python script

Open your text editor (TextWrangler or Notepad ++) and type the following lines:

In [179]:
```
print('This is my first python script')
math_answer = str(1+1)
print 'I can do math. 1+1=' + math_answer
print 'I can do math. 1+1={0}'.format(1+1)
```

```
This is my first python script
I can do math. 1+1=2
I can do math. 1+1=2
```