Minh	Isaac	BenE
Ben	Rey	Evan
Natalie	Derek	Chance
Nathan	Alex	Sienna

Victoria	Trevor	Chloe
Cameron	Ryan	Bitsy
Ryan	Sandra	Jude
David	Katja	Sam
		Alexis

Outline

Starter Programs

- feedback

Recursion

Text Manipulations

- string functions
- Exercises
- Secret message

Cool media art pieces

Prof. Angela Chang

Lecture 6: String Manipulations

Fall 2017.Sept 25

CODE, CULTURE, AND PRACTICE

Computing is cultural

Social, political, cultural

- Computers attempt to model the world, classifying status and dictate human behavior
  - history, cultural references, and assumptions
- Many ways to solve a problem
  - allowing for variations & innovations in ideas
- We choose what to program, how to explore it, and whether or not we wish to share and allow others to build
  - but we need to understand how computers think

Starter programs.ipynb

clarifications

Importing libraries gives you access to all the functions in that library.

Range and slicing have the same argument syntax
[start , end, step]

If/else statements
need to belong together

Order matters– put the most restrictive clause on top.

```
""" Ordering A Pizza That a Vegan Can Eat """

# things that a vegan does not eat
diet_restrictions = set(['meat','cheese'])

# decide which pizza to order
if 'meat' and 'cheese' in diet_restrictions:
    print('Get a vegan pizza.')

elif 'meat' in diet_restrictions:
    print('Get a cheese pizza.')

else:
    print('Get something else.')
```

Dictionaries, another way to select a value.

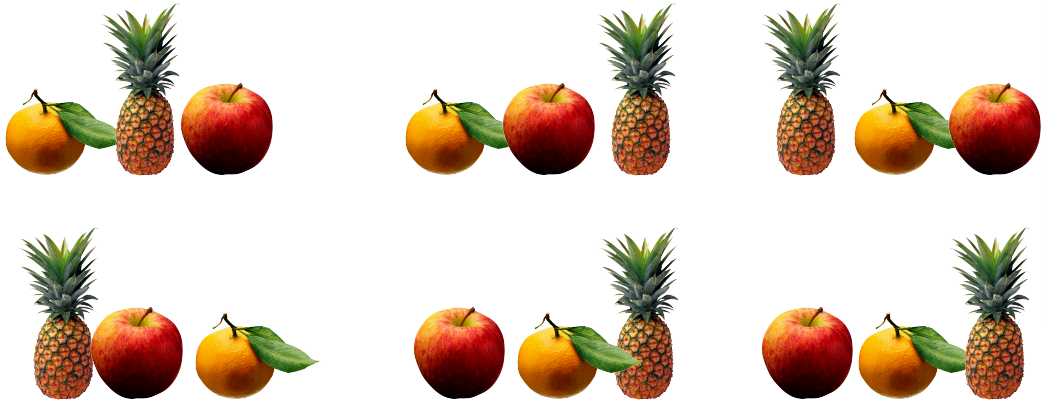
Factorial

number of arrangements for n items

- 1! =1

- 2! = 2 * 1 = 2

- 3! = 3 x 2 x 1 = 6



1 item can be arranged 1 way = [1]
2 items can be arranged in 2 ways:
2 * 1 [a,b] or [b,a]
3 items can be arranged in 6 ways:
[a,b,c], [a,c,b], [b,a,c]
[b,c,a], [c,a,b], [c,b,a]

```
def factorial(n):
    answer = 1
    if n < 0 :
        answer = 0
    for num in range(1,n+1):
        answer = answer * num
    return answer
```

Factorial- another way!

Factorial of n is equal to:

n times factorial(n-1)

4! = 4 x 3!

3 x 2!

2 x 1 !

1

```
factorial(4)
4 * factorial(3)
4 * 3 * factorial(2)
4 * 3 * 2 * factorial(1)
4 * 3 * 2 * 1
4 * 3 * 2
4 * 6
24
```

Recursive function:
Factorial referencing itself

```
def fact(n):
    if n == 0:
        return 1
    elif n<0:
        return 0
    else:
        return n * fact(n-1)
```

Recursion

Function references itself

Doubling every element using recursion

```
def doubler(sequence):
    if len(sequence) == 0:
        return[]
    else:
        return [ 2 * sequence[0] ] + doubler(sequence[1:])
```

if the length of the list is 0, return nothing, otherwise, return double the first element and the result of **doubler** on the rest of the list

```
[2*'a'] + doubler(['b','c'])
[2*'a'] + [2*'b'] + doubler(['c'])
[2*'a'] + [2*'b'] + [2*'c'] + []
[2*'a'] + [2*'b'] + ['2c']
[2*'a'] + ['2b'] + ['2c']
['2a'] + ['2b'] + ['2c']
['2a', '2b', '2c']
```

compare this to

```
def double(sequence):
    result = [];
    for element in sequence:
        result = result + [element*2]
    return result
```

double every element and save the results into a new list

Recursion

- identify base case
- identify repeating piece

```
def fact(n):
    if n == 0:
        return 1
    elif n<0:
        return 0
    else:
        return n * fact(n-1)
```

```
def double(sequence):
    result = [];
    for element in sequence:
        result = result + [element*2]
    return result
```

```
permutations(words)

if (len(words)) < 2:
    yield words
else:
    for i in range(len(words)):
        for result in permutations(words[:i] + words[i+1:]):
            yield [words[i]] + result
```

Permutation Poems Recursion.ipynb

Strings

Strings are a sequence of characters

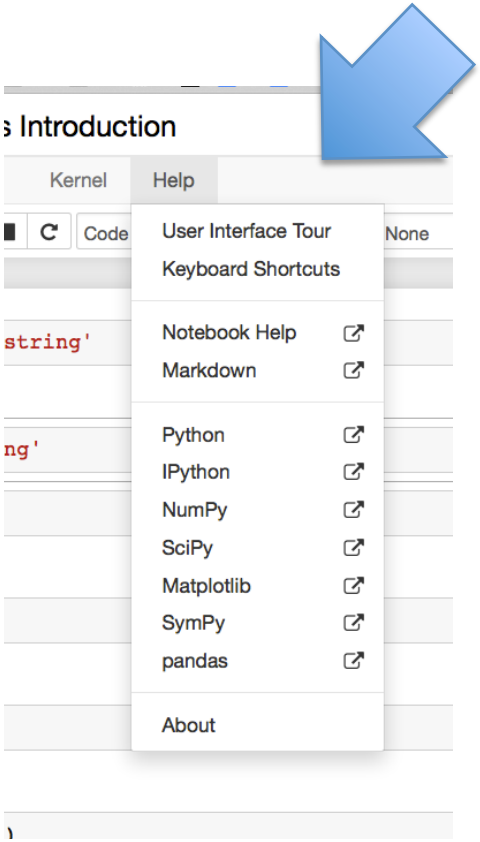
- Indicated with surrounding quotation marks “” or ‘ ‘ (must be matched)
- Can be assigned to variables
- Can be empty (null string, nothing between quotes)
- Are NOT numbers

Strings have *methods* which are very useful

- len(‘hello’), print ‘hello’
- Can get help through r
- look at python docs on string for modules
- can find the element using the index
  string[index]
- Can get substrings using slices
  [start: end : step]

```
len('2112')
```

```
4
```



Common string functions

join and split

```
names = ['Bob','Carol','Ted','Alice']
```

string_delimiter.**join**(list)

```
' & '.join(names)
```

You can also use the + operator.

```
for person in names:
    print person + " exists."
```

```
Bob exists.
Carol exists.
Ted exists.
Alice exists.
```

list.**split**(*string_delimiter*)

```
text = 'The quick brown fox.'
text.split(" ")
```

```
['The', 'quick', 'brown', 'fox.']
```

Sorting string lists

Two ways to sort:

sequence.sort() ---
sorts list in place, changes original

```
alist = ['c','a','b']
alist.sort()    #change orignal list
print alist
```

```
['a', 'b', 'c']
```

sorted(list) -
- **returns** the sorted list, leaves original

```
alist = ['c','a','b']
sorted(alist)  #original unchanged
print alist
```

```
['c', 'a', 'b']
```

Questions

How would you take every other character of a string of unknown length?

```
'abcdefgh'[::2]
```

How to you find the last character of a string?

How would you reverse a string of characters?

How do you split a string into a list of words?

How would you count the number of characters in a string?

Exercise: Detecting Doubles

Lets walk through a function…

Write a function that tells us how many double letters in a string.

> *They flee from me, that sometime did me seek*
>
> *With naked foot stalking in my chamber.*

From They Flee from Me by Thomas Wyatt

http://bit.ly/WyattFlee

How might we do this? Break it down….

1) Assign it to a variable to make it easy to refer to

2) Make all the letters lowercase

3) Iterate through the string and find all the double letters..

4) Count all the double letters.

DoubleLetters.ipynb

Exercise: Detecting Doubles

Write a function that tells us how many numbers in a string.

> *They flee from me, that sometime did me seek*
> *With naked foot stalking in my chamber.*

From They Flee from Me by Thomas Wyatt

http://bit.ly/WyattFlee

How might we do this?

1) Assign it to a variable to make it easy to refer to
2) Make all the letters lowercase
3) Iterate through the string and find all the double letters..
4) Count all the double letters.

Simplification: Count all ‘ee’ s

Exercise: Detecting Doubles

Write a function that tells us how many numbers in a string.

> *They flee from me, that sometime did me seek*
> *With naked foot stalking in my chamber.*

From They Flee from Me by Thomas Wyatt

http://bit.ly/WyattFlee

1) Assign it to a variable to make it easy to refer to

```
wyatt = 'They flee from me that sometime did we seek
/ With naked foot, stalking in my chamber.
```

2) Make all the letters lowercase

‘Oolong’.lower() ↵
‘HELLO World’.lower()↵
original = ‘Burma Shave’ ↵
original.lower() ↵

```
wyatt.lower()
```

Exercise: Detecting Doubles

….continued

```
wyatt = wyatt.lower() #overwrite the original variable with lowercase
```

wyatt↵

3) Iterate through the string and find all the double letters..

3a) Iterate through the string.

3b) Find all the double letters.

```
for c in wyatt:
    print c
```

Exercise: Detecting Doubles

….continued

```
wyatt = wyatt.lower() #overwrite the original variable with lowercase
```

3) Iterate through the string and find all the double letters..

3a) Iterate through the string.

```
for c in wyatt:
    print c
```

3b) Find all the double letters.

```
(len(wyatt)):
```

```
range(len(wyatt)):
```

```
for i in range(len(wyatt)):
    print wyatt[i]
```

```
for i in range(len(wyatt)):
    print i, wyatt[i]
```

Exercise: Detecting Doubles

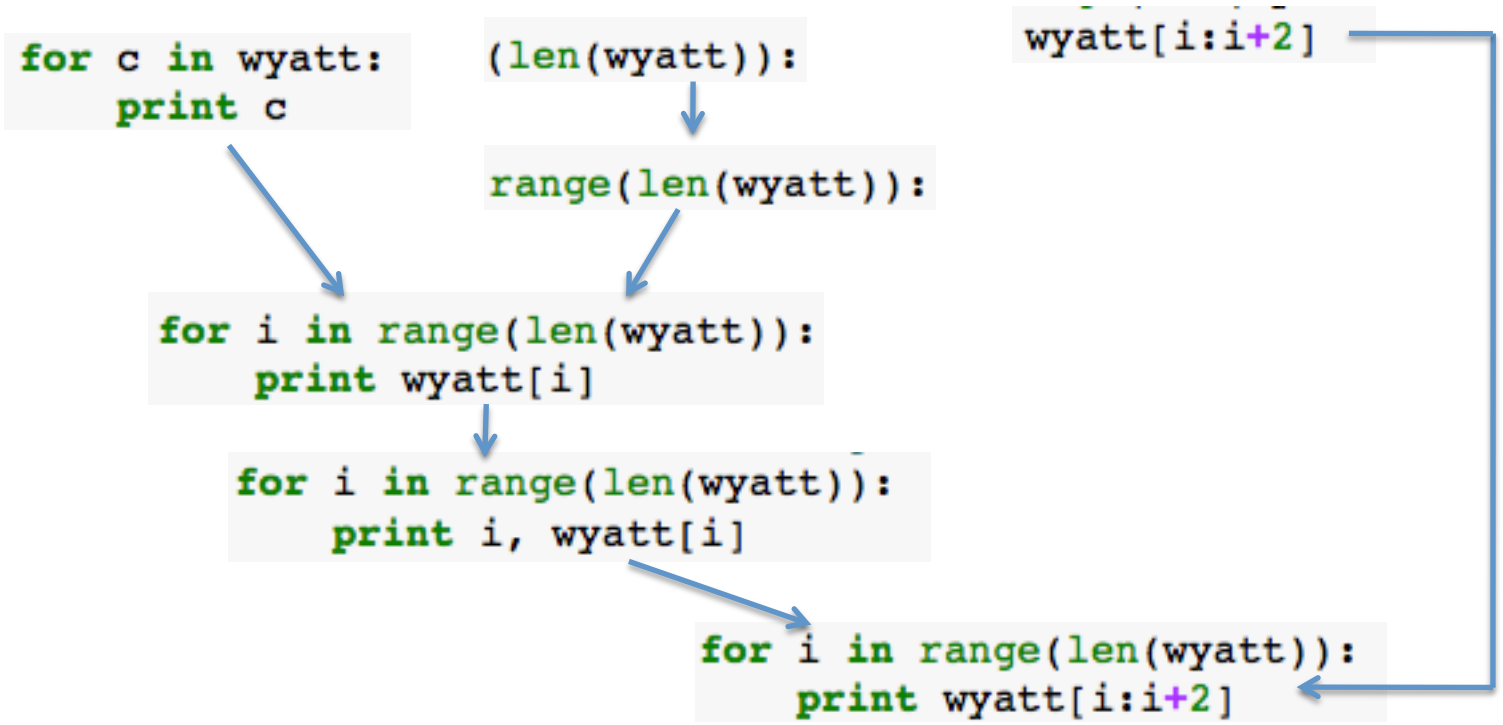
….continued

```
wyatt = wyatt.lower() #overwrite the original variable with lowercase
```

3) Iterate through the string and find all the double letters..

3a) Iterate through the string.

3b) Find all the double letters.



Exercise: Detecting Doubles

….continued

```
wyatt = wyatt.lower() #overwrite the original variable with lowercase
```

4) Count how many of them are ‘ee’

```
pair = 0
for i in range(len(wyatt)):
    print wyatt[i:i+2]
pair = pair + 1
```

```
pair = 0
for i in range(len(wyatt)):
    if wyatt[i]==wyatt[i+2]:
        pair = pair + 1
        print i,wyatt[i], wyatt[i+1]
```

Solve that index error!
Then bundle it into a function **twin()**

Group Exercise

Message Encoding/Decoding Exercise

Using your new knowledge of string indexing, slicing, splitting. and joining

Each person in the group:

- Decide on a message to encode.
- Write a function that takes the message string and encodes it in some way.
- Give your function to another person in the group.
- Have the second person write the decoder.
- Share your work with the class.

Deliverables:
Each person will submit
an ipython notebook with

1. A message
2. An encoder
3. A decoder
4. The names of the people who wrote each part

See for examples:
Message manipulation.ipynb

Upload your answer (work in progress) at:
for class participation credit

Summary

For next class

- Work through some of the notebooks on your own
- Upload the class participation