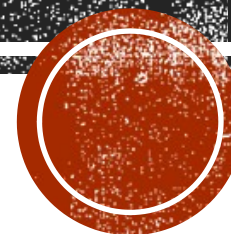


STRINGS



REVIEW: UNKNOWN NUMBER OF PARAMETERS

```
def add(*params):  
    total = 0  
    for i in params:  
        total += i  
    return total
```

```
>>> add(1,2,4)
```

```
>>> add(1,2,3,5,4,3)
```



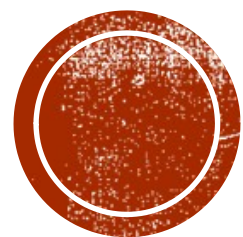
REVIEW: RECURSION



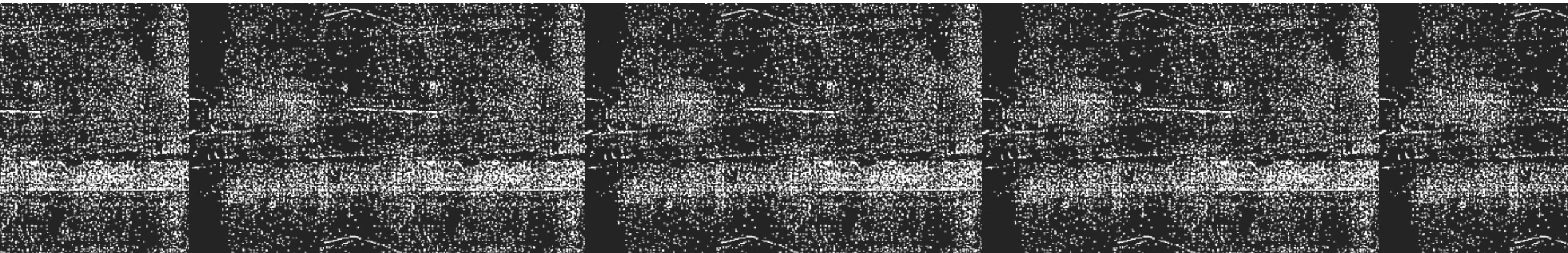
LOOPING AND COUNTING

Write a function that can determine the number of occurrence of given letter inside a string?





SEARCH PROBLEM



SEARCHING INSIDE A STRING

Find the first occurrence of a
given letter inside a string



STRING METHODS

A method is similar to a function

It takes arguments and returns a value but the syntax is different.



EXAMPLES

```
>>> word = 'test'
```

```
>>> new_word = word.lower()
```

```
>>> print(new_word)
```

```
>>> print(word)
```



EXAMPLES

```
>>> word = 'test test'
```

```
>>> i = word.find('s')
```

```
>>> print(i)
```



EXAMPLES

```
>>> word = 'test test'
```

```
>>> i = word.find('es')
```

```
>>> print(i)
```



EXAMPLES

```
>>> word = 'test test'
```

```
>>> i = word.find('es', 3)
```

```
>>> print(i)
```



EXAMPLES

```
>>> word = 'test test'
```

```
>>> i = word.find('t', 1, 3)
```

```
>>> print(i)
```



THE in OPERATOR

>>> 'n' in 'banana'

True

>>> 'seed' in 'banana'

False



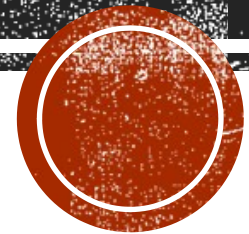
EXAMPLE

Write a function that can determine letters that are shared by two words?



DATA STRUCTURES

Data Structures In Python



WHAT ARE DATA STRUCTURES?

In computer science, a **data structure** is a particular way of **organizing data** in a computer so that it can be used efficiently.



LISTS

- A List is a kind of Collection
- A collection allows us to put many values in a single “variable”
- A collection is nice because we can carry many values around in one convenient package.



LIST

```
friends_cast = ['Chandler', 'Phoebe',  
                'Joey']
```

```
marks = [12, 15, 9, 12, 20, 19]
```



LIST CONSTANTS

- List constants are surrounded by square brackets and the elements in the list are separated by commas.
- A list element can be any Python object - even another list
- A list can be empty



EXAMPLES

```
>>> print([1, 24, 76] )
```

```
[1, 24, 76.0]
```

```
>>> print(['red', 'yellow', 'blue'] )
```

```
['red', 'yellow', 'blue']
```



EXAMPLES

```
>>> print([ 1, [5, 6], 7] )
```

```
[1, [5, 6], 7]
```

```
>>> print( [] )
```

```
[]
```



GO THROUGH THE LIST

```
for i in [5, 4, 3, 2, 1] :
```

```
    print(i)
```

```
print('Blastoff!')
```

5

4

3

2

1

Blastoff!





LIST TRAVERSAL

```
Simpsons= ['Homer', 'Bart', 'Lisa',  
           'Marge']
```

```
for character in Simpsons:  
    print(character, 'is in the Simpsons  
family')
```



HOW LONG IS A LIST?

- The `len()` function takes a list as a parameter and returns the number of elements in the list
- Actually `len()` tells us the number of elements of any set or sequence (i.e. such as a string...)



LIST TRAVERSAL

```
car_makers = ['Ford', 'Toyota',  
              'BMW']
```

```
for car in range(len(car_makers)):  
    print(car_makers[car])
```



THE in OPERATOR

```
>>> 'Ford' in ['Ford', 'Toyota',  
'BMW']
```

True



LIST OPERATIONS

>>> a = [1, 2, 3]

>>> b = [4, 5, 6]

>>> c = a + b

>>> c

[1, 2, 3, 4, 5, 6]



LIST OPERATIONS

>>> [0] * 4

[0, 0, 0, 0]

>>> [1, 2, 3] * 3

[1, 2, 3, 1, 2, 3, 1, 2, 3]



LIST OPERATIONS

```
>>> a_list = ["a", "b", "c", "d", "e", "f"]
```

```
>>> a_list[1:3]
```

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

```
>>> a_list[:4]
```

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



LOOKING INSIDE LISTS

Just like strings, we can get at any single element in a list using an index specified in **square brackets**

'Homer'	'Bart'	'Lisa'	'Marge'
,			,
0	1	2	3



UPDATING LISTS

```
friends_cast = ['Chandler', 'Phoebe',  
                'Joey']
```

```
friends_cast[0] = 'Monica'
```

```
print(friends_cast)
```



MUTABILITY: LIST VS STRINGS

Lists are mutable

Strings are immutable



STRING

```
>>> my_string = "TEST"
```

```
>>> my_string[2] = "X"
```

Traceback (most recent call last):

File "<pyshell#2>", line 1, in <module>

my_string[2] = 'x'

TypeError: 'str' object does not support item assignment



DEEP COPY & SHALLOW COPY

```
>>> game_of_thrones = ['Targaryen', 'Stark']  
>>> got_families = game_of_thrones  
>>> got_families[0] = 'Baratheon'  
>>> print(game_of_thrones)
```

