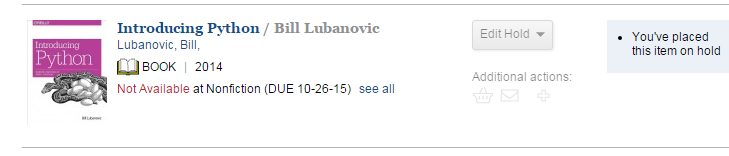
# Learning Python

Book: Introducing Python; Sunnyvale public library eBook.



## Interactive Interpreter:

Typing python on the command prompt will launch the python interactive interpreter which can perform most python operations.

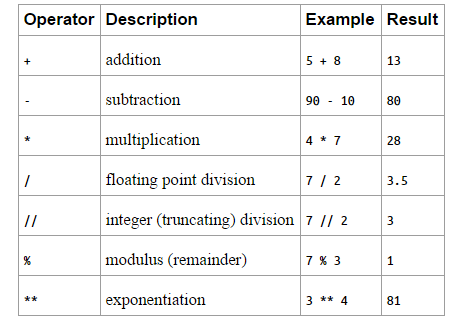
## Data Type

Python data type are strongly typed, meaning even if the variables declare are mutable ie their value can be changed, the data type of the variable cannot be changed.

## Variable Names

1. Can only contain uppercase or lowercase characters, digits and underscore.
2. Cannot begin with digit.
3. Variable starting with underscore have a separate interpretation.

## Mathematical operations in Python:



## Strings

Double quotes and single quote either of them can be used to define strings.

Triple single or double quotes are used to create strings of multiple lines.

Strings are immutable:

Name = ‘Henny’

Name[0] = ‘P’

Is not allowed in python instead we can use a replace or slice function.

Ie Name.replace(‘H’,’P’) or

‘P’ +name[1:] 🡪 will output Penny

Slice[start:end:step]

[:] all characters or items in the sequence

[start:] starts from the mentioned start value

[:end] from the beginning to the mentioned end value-1

[start:end] from the mentioned start to the mentioned end-1.

[start:end:step] from the mentioned end-1 while skipping the step number of items.

Other string functions: Len, split,join

Str.startswith(word)

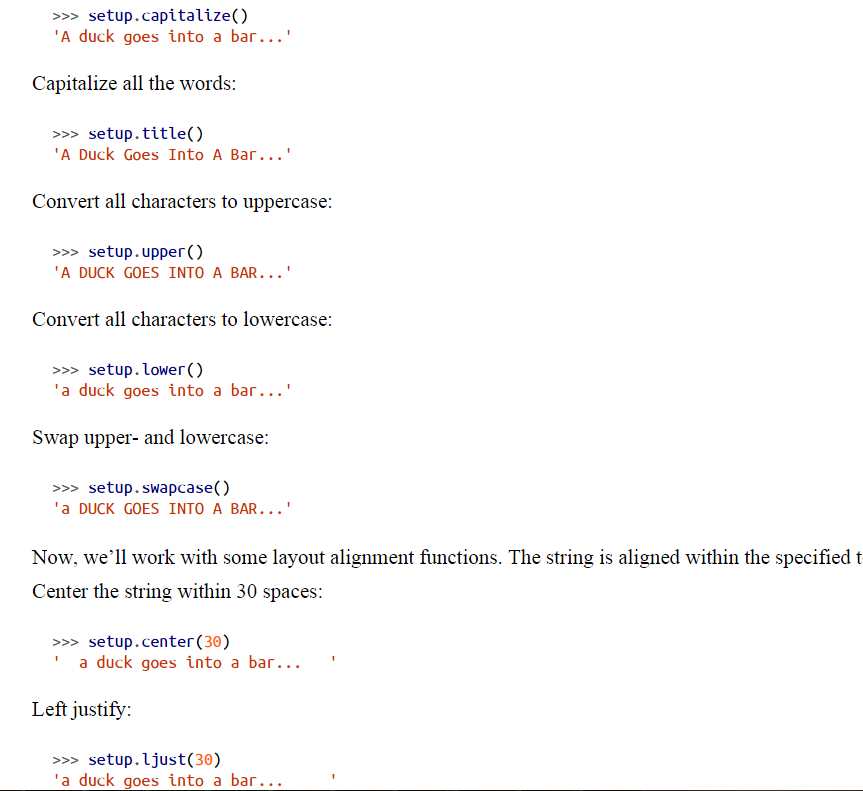
Str.endswith(word)

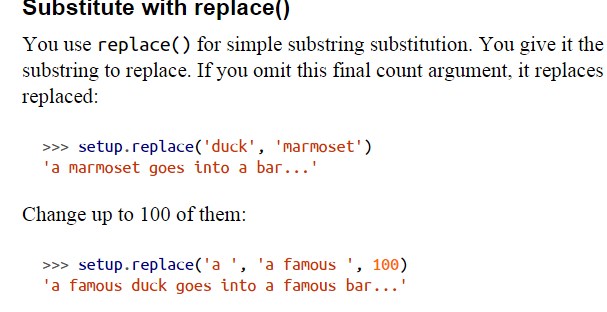
Str.find(word)

Str.rfind(word)

Str.count(word)

Str.strip(‘,’)





### Escape Character:

\ is used as an escape character to skip the interpretation of the character succeeded by it.

## Lists and Tuples

Lists and Tuples are sequences same as strings.

List is mutable, ie you can insert and delete elements in a List.

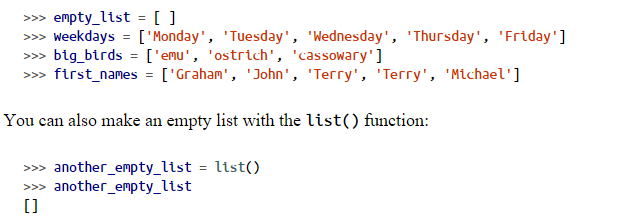
Tuple is immutable

### Lists:

Useful for keeping track of elements in order.

Unlike strings, Lists are mutable, you can change list in a place, add new elements and delete or over-write existing elements.

#### Creating Lists:



#### Convert Data Types to List using List()

List(‘cat’) will return a list of c , a ,t.

Atuple = (‘ready’,’fire’,’aim’)

List(Atuple); will translate the tuple into a list [‘ready’,’fire’,’aim’]

Split() is a variant of the list() function and chops a string into List based on a defined character.

List can contain lists as elements.

#### Slicing A List:

List[0:2] :- will select the first 2 elements of the list.

List[ : :2] – Will select all elements from start to end, and skip every other element.

List[ : : -1] :- Will select all elements from end to start and skip to the next element, essentially prints the list in reverse.

#### Append()

List.Append(elementn) will append the element at the end of the list.

#### Extend or +=

Combine or merge (not append) lists with List.Extend(list1) or List +=list1

#### Insert and Delete

List elements can be inserted or deleted using functions.

List.insert(offsetnumber,value)

Del List[offsetvalue]

#### Remove Method

List.Remove(element\_value) : deletes a specified element from the list.

#### Pop()

List.Pop(offset): This method pops the element in the mentioned offset from the list, default value is -1 or the last element in the list. This function returns and deletes the element specified.

#### Index()

List.index(value) : returns the index at which the element exists in the list.

#### In

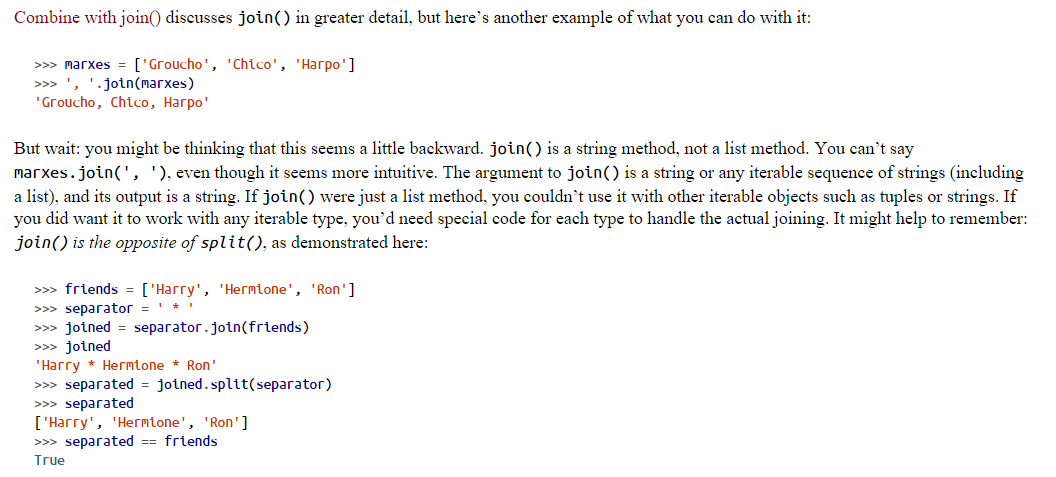
Pythonic way of checking if a certain value exists in the list.

Value in list -- returns True or False based on the existence of the value.

#### Count()

List.Count(element\_value) – returns number of occurrences of the element\_value in the list.

#### Join



#### Sorting Lists:

List.Sort() – sorts the list itself in place.

Sorted(list) – returns a sorted copy of the list.

#### Len:

Len(list) – returns the number of elements in the list.

Assigning New Lists:

List a, List b.

A= b, will assign b to a, but changing b will also change a and vice versa.

B = a.Copy(), will create a copy of list a and assign it to b. B and A will be independent lists.

### Tuple:

#### Creating tuples:

Tuple = () – an empty tuple is created.

One\_max = ‘Akbar’, - creates a tuple with one element in it.

Max = ‘Akbar’,’Birbal’,’Jodha’ – creates a tuple with 3 elements in it.

#### Tuple Unpacking:

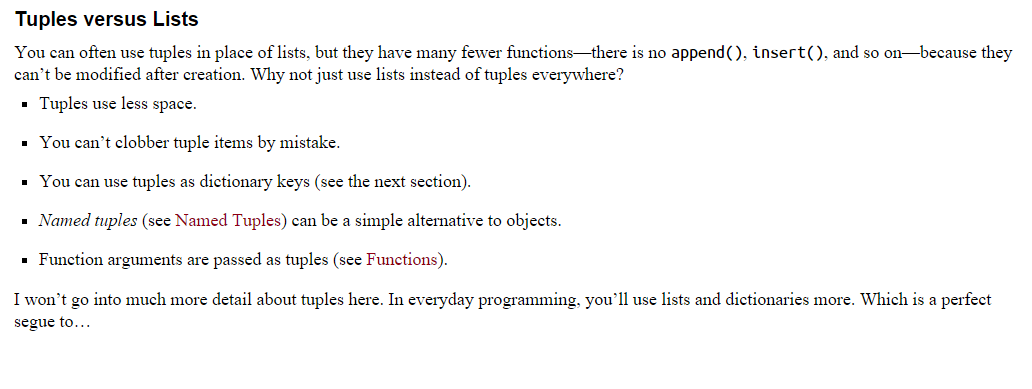
Multiple variable assignment with tuples:

A,b,c = max – assigns Akbar into A, Birbal into b and Jodha into c.

#### Tuple Conversion function:

A = Tuple(list) – converts a list into a tuple.

#### Tuples vs Lists



## Dictionaries:

Dictionary: is similar to a list, order is not important, and elements are not selected by offset, instead elements are selected by key value pairs.

The key is often a string, but it can be any ***immutable*** data type in python ie Boolean, float, integer, string or a tuple.

Dictionaries are mutable, so you can add delete and change their key-value elements.

#### Creating a Dictionary

DictiA = {} – creates an empty dictionary.

DictiB = {

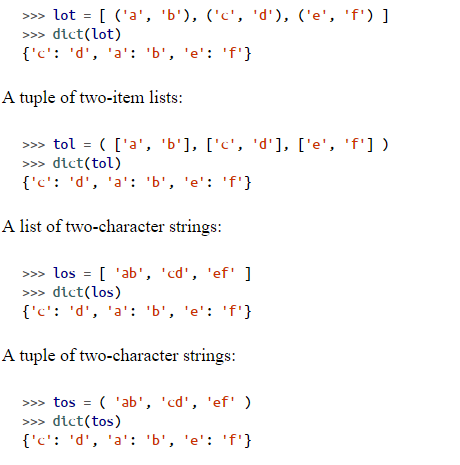
‘Key1’:’Value1’

…

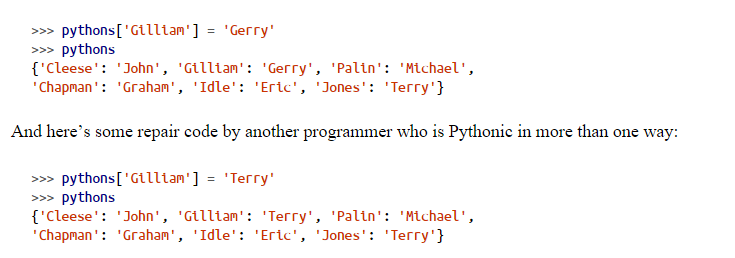
}

#### Dict()

Using Dict, we can convert two value sequences into a dictionary, in such sequence the first element is used as key and the second is used as value.



Adding a Key Value into a dictionary:



#### Update()

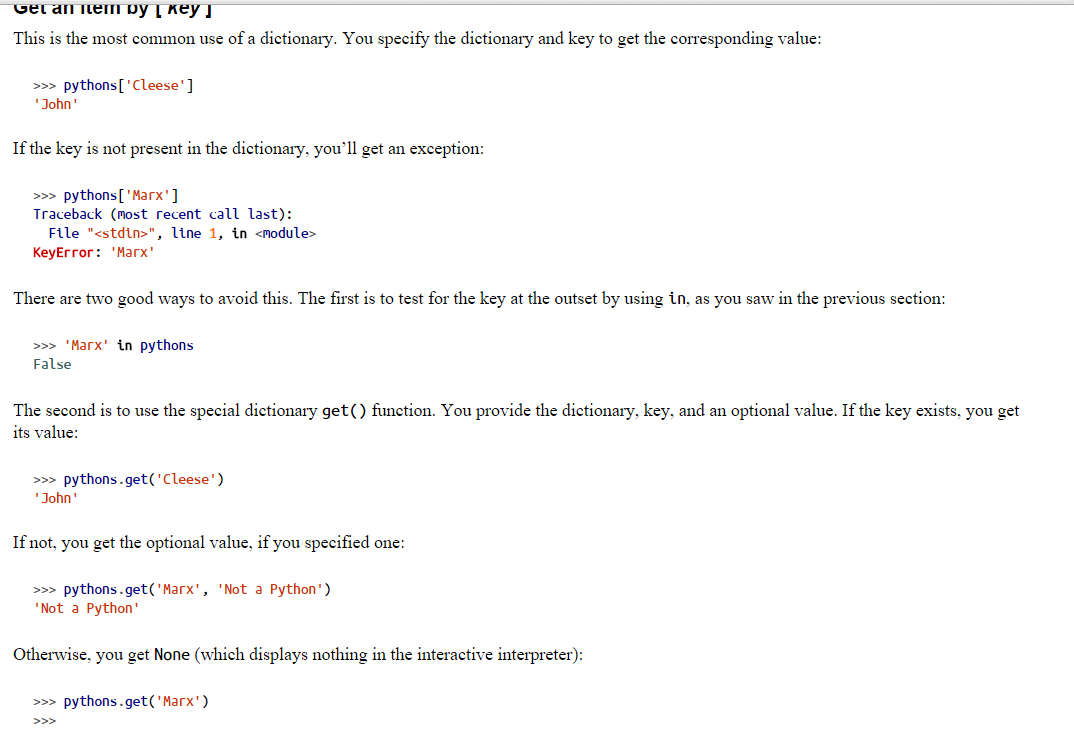
Dictionary1.Update(Dictionary2) – This will add the key value pairs in dictionary2 into dictionary1, if a key exists in both, Dictinoary2 wins.

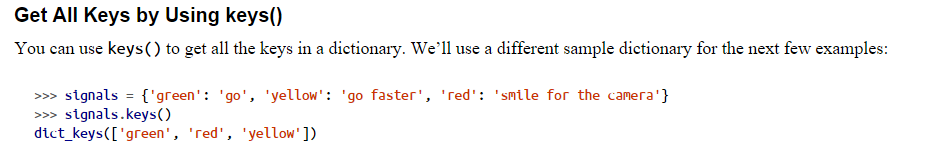
#### Deleting Keys

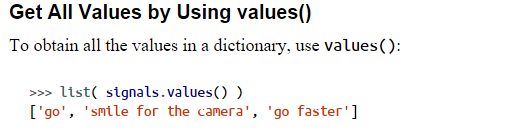
Del Dictionary[‘element\_value’]

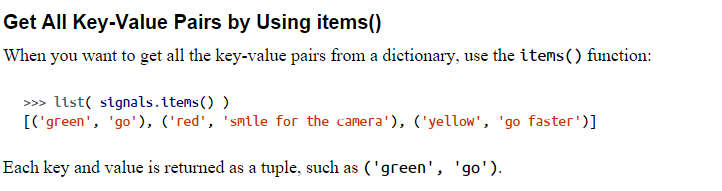
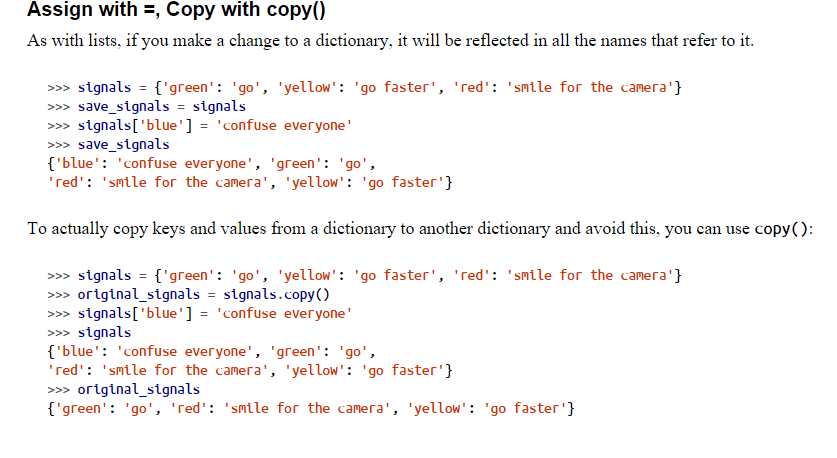
Dictionary.clear() – delete all elements.

Reading an element in Python





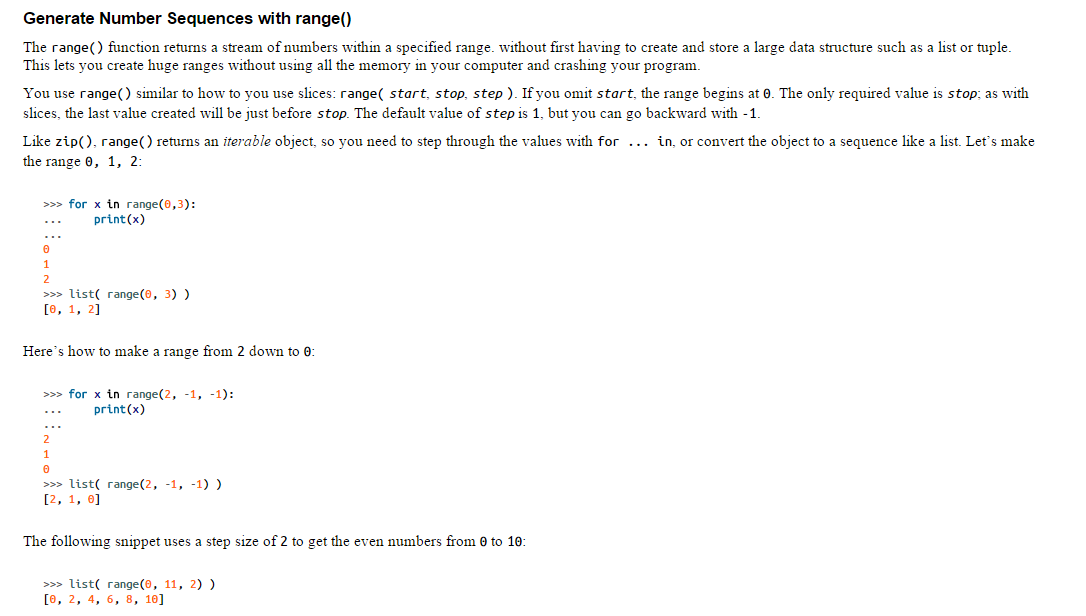




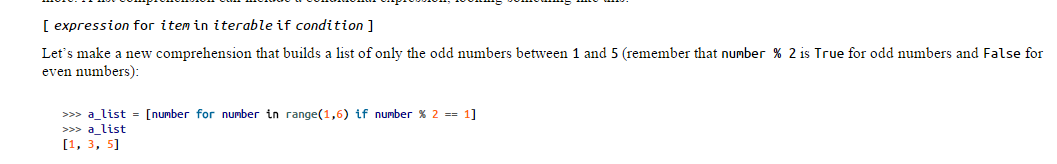
#### ZIP()



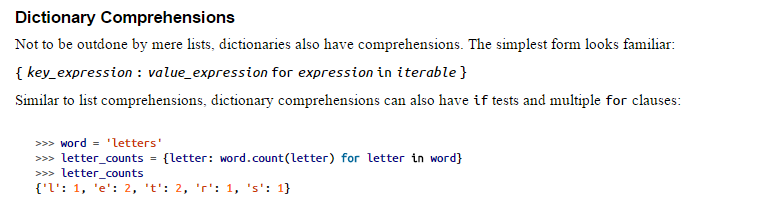
#### Range



#### Comprehension



#### Dictionary Comprehensions



#### Set Comprehensions

