

Big Daddy's

Data Container Quick Chart

Dictionaries (map type, mutable)		
Action/Event	Key Term/Symbol/Method/Attribute	Example
Create initially	{key:value, key:value,} Newdict = dict.fromkeys(keys [,values]) -	DPreps={"of":"coming from", "over":"on top of", "with":"accompanying"} Dprep2 = dict.fromkeys(LPreps,"hold") ~Note: no duplicate keys allowed~
Combine/merge containers	dict.update(dict2) ~adds dict2 to dict1~	D1.update(D2) ~also D1.update(1=dog, 2=cat)~
Combine overlap items only	N/A	
Combine non-overlap items only	N/A	
Add at first position	N/A	
Add items at end	N/A - dictionaries are not ordered	
Add one item at end	N/A - dictionaries are not ordered	
Add at position in container	N/A - dictionaries are not ordered	
Add somewhere inbetween	dictionary[key]=value	DPreps["at"]="close to"
Add multiple items	dict.update(dict2) ~adds dict2 to dict1~ or simple concatenation...	D1.update(D2) ~also D1.update(1=dog, 2=cat)~ ~not supported~
Remove a known value or key	del dictionary[key]	del DPreps["over"] ~view obj (list=keys) are dynamic!
Remove item(s) by index	N/A - dictionaries are not ordered	
Remove and return the last item	N/A - dictionaries are not ordered	
Remove and return a known item	pop(key[,default])	DPreps.pop("of")
Remove and return a random item	popitem()	DPreps.popitem()
Remove and return item number i	N/A - dictionaries are not ordered	
Replace an item/pair or value	~revalue based on key~	DPreps["with"]="possessing"
Replace a group of items	.update ~overwrites existing values~	D1.update(D2) ~also D1.update(1=dog, 2=cat)~
Retrieve sequential items	N/A - dictionaries are not ordered	
Retrieve values, keys, or pairs	dict.items(), dict.keys(), dict.values()	KeyList = list(DPreps.keys())
Retrieve value from known key	dictionary[key] or dictionary.get[key]	print(DPreps["of"]) ↗ "coming from"
Retrieve all keys, values, pairs	d.keys(), d.values(), d.items()	print(DPreps.keys()) ↗ dict_keys(['over', 'with', 'of'])
Retrieve index number of first value x	N/A - dictionaries are not ordered	
Compare overlap		
Compare subset		
*compare as true subset(not equal)		
Compare superset		
*compare as true superset(not equal)		
Iteration (loop)	TList = list(DPreps) ~ for i in range(0, len(TList)): ~ print(TList[i] + ":" + DPreps[TList[i]])	
Iteration (iter, next)	iter, next IT=iter(DPreps); For rec in DPreps: ~ xkey=(next(IT)); print(xkey) print(DPreps.get(xkey))	
Return number of items/pairs	len(dictionary)	len(DPreps)
Find count of x values	~can only hold unique keys - no duplicates~	
Find maximum value	N/A	
Find minimum value	N/A	
Determine membership	key in , key not in ~returns boolean value~	"of" in DPreps ↗ True "among" in DPreps ↗ False
-	.has_key(key)	DPreps.has_key("with")
Copy	dictionary.copy()	NewPreps = DPreps.copy()
Sort	N/A - dictionaries are not ordered	
Reverse items	N/A - dictionaries are not ordered	
Clear all	dictionary.clear()	DPreps.clear()
Delete the object	del dictionary	del DPreps
Convert		
Other: setdefault	dictionary.setdefault(key[,default]) ~if key in dict return value, if not insert with value of default~	